Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations | 1. Thesis and Dissertation Collection, all items

2012-03

Multi-Agent Task Negotiation Among UAVs to Defend Against Swarm Attacks

Day, Michael

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/6784

Downloaded from NPS Archive: Calhoun

http://www.nps.edu/library

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

MULTI-AGENT TASK NEGOTIATION AMONG UAVS TO DEFEND AGAINST SWARM ATTACKS

by

Michael Day

March 2012

Thesis Co-Advisors: Timothy H. Chung
Chris Darken

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704–0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–3–2012	Master's Thesis	2011-01-01–2012-03-30

4. TITLE AND SUBTITLE

Multi-Agent Task Negotiation Among UAVs to Defend Against Swarm Attacks

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Michael Day

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of the Navy

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A

14. ABSTRACT

This research involves a multi-agent based simulation modeling a large swarm of adversarial UAVs attacking a surface target and groups of friendly UAVs responding to thwart the attack. Defense systems need to cooperatively negotiate which enemy systems to engage to maximize the number of aggressor systems destroyed. Using optimal centralized task assignment methods as a baseline, various distributed methods are examined for efficiency and effectiveness. Our findings indicate that the optimality of distributed methods does approach that of centralized methods, though further study is warranted in future simulations with additional constraints, and in field experimentation with physical UAVs. We further find that the number of defender agents, the effectiveness of their weapon systems, and their speeds contribute significantly to the defender swarm's effectiveness.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified	UU	81	

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

MULTI-AGENT TASK NEGOTIATION AMONG UAVS TO DEFEND AGAINST SWARM ATTACKS

Michael Day
Civilian, Department of the Navy
B.S., University of Alabama Huntsvile, 2003

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 2012

Author: Michael Day

Approved by: Timothy H. Chung
Thesis Co-Advisor

Chris Darken
Thesis Co-Advisor

Professor Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This research involves a multi-agent based simulation modeling a large swarm of adversarial UAVs attacking a surface target and groups of friendly UAVs responding to thwart the attack. Defense systems need to cooperatively negotiate which enemy systems to engage to maximize the number of aggressor systems destroyed. Using optimal centralized task assignment methods as a baseline, various distributed methods are examined for efficiency and effectiveness. Our findings indicate that the optimality of distributed methods does approach that of centralized methods, though further study is warranted in future simulations with additional constraints, and in field experimentation with physical UAVs. We further find that the number of defender agents, the effectiveness of their weapon systems, and their speeds contribute significantly to the defender swarm's effectiveness.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

1 Introduction **1**
1.1 Motivation 1
1.2 Research Questions 2
1.3 Literature Review 2
1.4 Methodology 5

2 Assignment Algorithms **7**
2.1 Centralized Algorithm A - Basic Assignment 7
2.2 Centralized Algorithm B - Redundant Targeting 10
2.3 Decentralized Algorithm A - Implicit Coordination 12
2.4 Decentralized Algorithm B - Market-Based 12
2.5 Decentralized Algorithm C - Implicit Market-Based 14
2.6 Decentralized Algorithm D - Fixed Assignment 14
2.7 Assignment Persistence 14
2.8 Implementation 15

3 Experiment Design **19**
3.1 Response Variable: Percentage of Reds Destroyed 19
3.2 Factorial Design. 19
3.3 Factor Screening 28

4 Analysis **35**
4.1 Logistic Regression 35
4.2 Significant Factors 36
4.3 Prediction Model 39
4.4 Individual Factor Analysis. 43

5 Conclusions and Recommendations **49**
5.1 Conclusions 49

5.2 Recommendations 51
5.3 Future Work 51
5.4 Lessons Learned 52

6 Appendix **61**
6.1 GLPK Model File For Transportation Problem. 61

Initial Distribution List **63**

List of Figures

Figure 2.1 Leading red agents assigned to trailing blue groups. 8

Figure 2.2 Leading red agents assigned to leading blue groups. Note that if leading blue groups successfully engage their targets, the remaining members of those groups are available for reassignment. 9

Figure 2.3 Repast Simphony user interface. Parameters section is circled. 15

Figure 2.4 Two different views of blue agents in launch phase in the Repast Simphony visualization tool. Distant red agents can also be seen approaching in the background. The blue HVU is the cylinder in the foreground. . . 17

Figure 2.5 Agents in Repast Simphony. Blue agents on the left have not yet left launch phase. The other blue agents on the left have entered transit phase. Red agents are on the right side. 17

Figure 3.1 Red agent entering the secure perimeter of the blue HVU. 20

Figure 3.2 A red agent approaching the blue HVU. The blue agent must take off when the red is at a sufficient distant, S (or the Secure Perimeter Radius), so that the blue will reach point P before the red does. The values of the other variables are discussed in the text. 21

Figure 3.3 Blast range of warheads of representative anti-air missiles from open literature. 23

Figure 3.4 Lifetime of a blue agent. The end of the Transit Phase is determined by the Commit Range factor. Once a blue enters within that distance of its red target it ceases to update its assignment, stops flocking with its subteam, and enters Engage Phase. 24

Figure 3.5 Illustration of how dive angle is defined. 26

Figure 3.6 Various arrival angle ranges: (a) a 10° arrival angle range, (b) a 180° arrival angle range, (c) a 360° arrival angle range. 27

Figure 3.7 JMP screen for designing a D-Optimal experiment. 31

Figure 3.8 Asymptotic Standard Deviation for a single design point. 32

Figure 4.1 Least squares regression of data with the maximum adjusted R^2 calculated via JMP. 36

Figure 4.2 Least squares regression of data with adjusted R^2 of 0.90. Calculated and plotted via JMP. 37

Figure 4.3 Significant factors identified in order of significance in regression with adjusted R^2 of 0.90 38

Figure 4.4 A least squares mean study showing that different assignment methods do not vary greatly in how they affect the response variable. 38

Figure 4.5 Steps in the stepwise regression listed in JMP. Assignment method enters the model as a factor on step 13 and changes R^2 from 0.9015 to 0.9112, thus accounting for only 0.0097 of the variability in the model. 39

Figure 4.6 Fitting the residuals of the logit of the response variable reveals a near normal fit with mean close to zero. 39

Figure 4.7 No pattern perceived in a plot of residuals of the response variable. . . 40

Figure 4.8 Lack-of-fit test reveals there is curvature in our fit that we have not accounted for. 40

Figure 4.9 Top twelve parameter estimates (plus intercept) for model fitting provided by JMP. 41

Figure 4.10 Results of tests of the predictive model with the six design points of Table 4.1. 42

Figure 4.11 Predictions and simulations of percentage of reds destroyed when varying (a) number of blues per red, and (b) blue P_k. 43

Figure 4.12 Predictions and simulations of percentage of reds destroyed when varying (a) red speed, (b) blue speed, (c) number of reds, and (d) initial distance between reds. 44

Figure 4.13 Interactions of the number of reds factor with two other factors: (a) initial distance between reds does interact, but has only a slight effect, while (b) red transit speed interacts much more. Effects are discussed in greater detail in the text. 45

Figure 4.14 Interactions of the number of blues per red factor with two other factors: (a) red transit speed has a noticeable effect on the effect blues per red has on the response variable and (b) number of red subteams also has an effect. 45

Figure 4.15 Interaction of Blue P_k with red subswarm arrival angle range. 46

Figure 4.16 Interaction between red and blue speeds. 47

Figure 4.17 Interaction between blue max speed and dive angle. 48

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

Table 3.1 Experiment Design Factors 33

Table 4.1 Factor levels for the test design points used to verify the predictive model. All other factors were kept at a constant level. 41

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgements

I would like to thank my advisor Chris Darken for introducing me to agent-based modeling and always presenting it in a way that made me want to learn more. I thank my advisor Tim Chung for the many many hours spent teaching a variety of topics to me, too often past the hour he would have preferred to have been home. I thank Rachel Silvestrini for patiently walking me through the world of statistical analysis of experiments.

Most of all, I thank my wife Evie for enduring the life of a single parent as I struggled to complete my degree. She never let me quit and it is because of her that I was able to finish at all.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

1.1 Motivation

Unmanned systems are becoming a vital part of the U.S. military's arsenal. [1] states:

> By performing tasks such as surveillance; signals intelligence (SIGINT); precision target designation; mine detection; and chemical, biological, radiological, nuclear (CBRN) reconnaissance, unmanned systems have made key contributions to the Global War on Terror (GWOT). As of October 2008, coalition unmanned aircraft systems (UAS) (exclusive of hand-launched systems) have flown almost 500,000 flight hours in support of Operations Enduring Freedom and Iraqi Freedom, unmanned ground vehicles (UGVs) have conducted over 30,000 missions, detecting and/or neutralizing over 15,000 improvised explosive devices (IEDs), and unmanned maritime systems (UMSs) have provided security to ports.

Unmanned systems are not limited to the U.S. military. International governments are also actively developing and deploying these systems, including potential adversaries such as Iran and China. Iran has a UAV bomber, Karrar (Farsi for "Striker"), that can deliver a 500 lb bomb or a short range cruise missile 300 miles [2]. China has actively pursued anti-radiation autonomous UAV systems which have been identified as a "significant threat to various critical military C4ISR facilities on Taiwan as well as to U.S. operational forces in the region" [3]. These attacks involve swarms of UAVs attacking single high value targets and overwhelming their defence systems by the sheer number of UAVs involved in the attack.

These kinds of attacks can be envisioned on land based radar or other targets. In fact, these attacks need not come exclusively from unmanned aerial systems. Unmanned surface, ground, and undersea systems may conceivably be employed as well. Unmanned systems also need not be lethal; swarms of systems could be employed for misdirection, jamming, and other battlefield requirements. As the use of unmanned systems accelerates, it is easy to envision scenarios in which unmanned systems engage other unmanned systems.

Designing groups of unmanned systems capable of engaging other groups of unmanned systems is more cost effective when modeling and simulation is employed during system development. Systems can be modeled with some parameters of interest in mind. Simulations can be performed using the models and many different possible system configurations can be examined without having to physically build and test those systems. Given a set of objectives, an ideal set of parameters can be found through experimentation and analysis [4].

1.2 Research Questions

This work examines methods of assigning blue defence UAVs to aggressor red UAVs in order to protect a high value asset owned by the blue team. There are centralized assignment methods that are known to be optimal, but non-optimal decentralized methods exist that are less computationally complex (see Section 1.3). Also, individual agents might have access to global or local information, e.g., sensing data for an individual agent is not global in a completely decentralized solution. The research questions we wish to explore are:

1. Do decentralized assignment methods approach centralized methods' effectiveness in complex scenarios with a large number (up to 150 aggressor vs. 450 defender) of agents?

2. How much do other factors in a complex scenario contribute to a blue team's defence capability as compared to the assignment method (i.e., is assignment method a major factor in blue's effectiveness)?

3. What are the tradeoffs between global and local information availability versus performance?

1.3 Literature Review

Studies involving large groups of cooperative agents have been conducted as in [5, 6, 7], but these studies are based on cellular or insect behavior and the individual agents are not as complex as a UAV system. Studies do exist in which small groups of more complex unmanned systems cooperate with or compete against one another, but further study needs to be done for larger group sizes. The work of [8] studies coordination of tasks and control of groups of three to six UAVs. Jin and Polycarpou [9] study teams of UAVs performing search and destroy missions with battle damage assessment, having teams of size two to ten. A study in [10] examines

the problem of routing UAVs to predetermined fixed targets with group sizes four to seven. This work seeks to model the behavior of a large number of agents who are individually complex.

Models are developed for both red and blue systems, leveraging previous work as in [11]. A good basis for both the UAV models (aggressor and defender) is utility-based agents. Agents are "anything that can be viewed as perceiving its environment through sensors and acting upon that environment" [12]. Utility-based agents have a goal and employ a utility function to map the degree to which the goal is being met to a real number. The goal of the blue agents in this work is to minimize the total distance to be traveled by the blue team and this is periodically checked via a matrix of Euclidean distances of individual blues to individual reds maintained throughout the scenario (see Chapter 2). Previous work has had UAVs search for stationary targets using distance as a measure of utility [13].

Though scenarios can be envisioned where autonomous sea, land, or air unmanned systems may act in groups in order to achieve a goal or attack a specific target, this study confines itself to unmanned air systems. The study examines countermeasures for swarms of UAVs performing a SEADS, that is, suppression of enemy air defense systems, attack against a surface target. A major objective of the study is to see how various task assignment methodologies protect the surface target. It has been observed that it is difficult to find a one-size-fits-all coordination approach [14], so we use a domain-specific approach.

The scenario outlined in Section 1.4 is based on a review of various existing physical systems. It is possible that more capable (i.e., faster, higher endurance) systems exist, but information on their specifications or existence is difficult to locate in open literature. It is likely that our scenario and simulator can be scaled to reflect values required for many different systems and the predictive model proposed in Section 4.3 provides conservative estimates for red attrition rates for a wide variety of scenarios. We choose a maximum of 150 incoming reds because an *Arleigh Burke* class destroyer as it has about 90 surface-to-air missiles [15] and we assume that many high value assets would have comparable defences. The range of blue UAV weapon systems were based on [11, 16, 17, 18] and a range of 10 meters was chosen for our scenario. However, it is probable that weapon systems exist that have higher ranges since lethality specifications on more modern systems are difficult to find in open literature. Choosing the P_k for blue weapon systems also proved difficult based on what is available in open literature. We used some data on anti-air artillery in [19] to arrive at a conservative estimate of 0.85 for P_k for a given blue's weapon system. We chose to vary P_k in order to account for variability in available weapon

systems (see 3.2.5). The choices of near 12,000 ft as a cruising altitude for incoming reds as well as speeds varying between 80 and 160 knots for both blue and red UAVs were based on a study of specifications of Harpy and similar systems [20, 21, 22, 23]. Red arrivals can be modeled as a Poisson process since arrivals times between red subteams can be assumed to be independent [24] – blue team does not know when reds will arrive .

In order for subteams of blues to behave as a cohesive unit, methods of keeping them spatially close were examined. Classic Boids flocking [25] mimics the behavior of flocks of birds and schools of fish and can be characterized by three parameters: (1) cohesion, or the degree to which individual flock members seek to remain spatially close, (2) separation, or the degree to which individual members seek to avoid colliding with one another, and (3) alignment, or the degree to which individual flock members seek to move in the same direction. Other methods of maintaining spatial cohesion include collective potentials [26], and a variation of Boids flocking that incorporates inertia [27]. A comprehensive survey of flocking schemes was performed in 2010 by [28]. We use classic Boids flocking for this study due to its simplicity and the existence of source code libraries that already implement it.

Centralized solutions to the task assignment problem have been explored rather extensively. In [29] a centralized method of assigning prioritized tasks with deadlines is discussed, but does not address the optimality of their solution. Another study, [30] deals with task assignment in a cooperative transport context and uses mixed integer linear programming as a means of achieving optimality. Gerkey lays out a taxonomy of task allocation methods [31] and cites mixed integer linear programming as one way to optimally solve task assignment and other problems. This study also use linear programming as a baseline to learn what the optimal solution to our assignment problem is (see Sections 2.1 and 2.2).

As for decentralized solutions, Alighanbari, et. al. point to a method that involves performing the optimal centralized solution on each UAV, given the assumption that those agents have close to global situational awareness [32, 33, 34]. Global situational awareness is an assumption that can easily be made in simulation, but in operational settings accurate global information is nearly impossible to obtain [35]. Alighanbari acknowledges this in his work, exploring means of communication between agents to reduce each individual agent's lack of global situational awareness. Work in [36] explores a similar method of maintaining situational awareness via communication and coordination between individual agents in the context of robot soccer. Beard and Stepanyan study the feasibility of generalizing communication and coordination

among many types of vehicles, be they air, land, or sea based [37]. Further work by Olfati-Saber and Murray seek to alleviate problems caused in typical operational networks with switching and time delays [38].

Market based solutions, though not optimal, have been used in to overcome the computational complexity that tends to come with mixed integer linear programming solutions. These methods are based on economic models [39]. Auction methods of task coordination also attempt to deal with agents dealing with noisy, dynamic environments, with no a priori assignment [40, 41]. Work in [42] uses distributed auctions as a means of dealing with communication delays.

This work uses an auction algorithm to solve the centralized case with less computational requirements than the optimal mixed integer linear program solution. For a decentralized method, we combine the idea of performing the centralized solution to all agents with a market based solution: the centralized algorithm they perform is not a mixed integer linear program but an auction, thus distributing the auction (see Section 2.5).

1.4 Methodology

The scenario that is played out is a classic red vs. blue simulation. Red agents seek to engage a high value blue target. Blue agents attempt to engage and kill red agents before they are able to reach the blue high value target. The experiment is outlined here, but is given in greater detail in Chapter 3.

1.4.1 Red Strategy

Aggressor UAVs can be modeled as utility-based agents whose goal is to strike a target. A group (or subteam) of red agents arrives according to a Poisson Process and attack using variations on the following strategies:

1. Each individual in the red team arrives simultaneously and the swarm is tightly packed.

2. Each individual in the team arrives simultaneously and the swarm is more loosely packed. The degree of tightness is a parameter that can be set.

3. The entire red team does not arrive at once. Individual subteams arrive that are loosely or tightly packed. Number of subteams and mean number of members of a sub-swarm are parameters of the simulation model.

4. The entire red team does not arrive at once, nor does not arrive as subteams. Rather, individual UAVs arrive at independent arrival rates.

All red strategies involve dive bombing the surface target from high altitude. This is so the red swarm is approaching at the highest possible speed when it punctures the surface vessel's anti-aircraft radar. The dive angle of individual UAVs is a model parameter of the simulation model.

1.4.2 Blue Strategy

Individual defenders can also be modeled as utility-based agents, but they cannot simply have the goal of striking an aggressor because the entire defensive swarm cannot greedily attack the first few aggressors and then be useless during the remainder of the attack (we assume blue agents employ a single use weapon). Aggressor targets must be given to each defensive agent and that requires some form of task assignment. The baseline strategy is a variation of a centralized solution to the optimal assignment problem [31]. Distributed task assignment strategies are compared against the centralized baseline solution.

1.4.3 Simulator

All UAV agents are relatively low fidelity. Flight dynamics are only roughly modeled; model parameters of more interest are assignment method, speed, probability of kill, number of agents per swarm, arrival of swarms, and tightness of swarms. The measure of performance is the attrition percentage of red untis. Agent models are built and tested using the Repast Symphony simulator [43]. This is a Java based toolkit, which allows us to do data collection using custom Java programming. The toolkit allows for visualization of single runs and also Monte Carlo runs to test parameters of interest and perform the experiments.

This study takes the red group's attack strategy as a given even though the red agents may attack using a variety of strategies. Future studies may employ machine learning algorithms to determine what strategy red is using as this would more closely model an actual attack scenario. This study also employs intentional cooperation as in [31]; comparing intentional versus emergent cooperative approaches may be a line of future research.

CHAPTER 2:
Assignment Algorithms

Task assignment of blue agents to red agents is performed in this study with the with the goal of completely defending a blue high value asset from attack by the red agents. In order to perform effectively, task assignment methods employ some notion of cost in order to measure the degree to which the goal is being met. We use Euclidean distance of red agents to blue agents as a measure of cost in this study. A survey of methods is presented beginning with algorithms assuming perfect information from perfect sensors and communications, and moving toward algorithms that make fewer assumptions but become necessarily more sophisticated. The survey is divided into centralized and decentralized algorithms. All centralized methods suffer from the need for a centralized oracle that has near perfect situational awareness of the entire scenario at all times and near unlimited bandwidth to communicate with all of its assets. Decentralized methods seek to remove those two constraints while striving to maintain solutions that approach optimal solutions otherwise found by centralized algorithms.

2.1 Centralized Algorithm A - Basic Assignment

The classic assignment problem (also called the linear assignment problem to differentiate it from a different problem: the quadratic assignment problem) deals with the problem of assigning n agents to n tasks. Agents incur some cost when being assigned to a task, and any agent can be assigned to any task. Formally, an $n \times n$ cost matrix, C, where $C_{i,j}$ denotes the cost of assigning agent *i* to task *j*. X is the $n \times n$ binary matrix which denotes assignments of n agents to n tasks and comprises the decision variables in the following linear program:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} C_{i,j}X_{i,j} \\ \text{s.t.} \quad & \sum_{j=1}^{n} X_{i,j} = 1 \text{ for } i = 1 \dots n \\ & \sum_{i=1}^{n} X_{i,j} = 1 \text{ for } j = 1 \dots n \\ & X_{i,j} \in \{0,1\} \text{ for all } i,j \end{aligned} \tag{2.1.1}$$

In this formulation we seek to minimize to total cost to all agents. The assignment problem is well known [44, 45, 46] and has been studied as early as the 19th century [47]. Methods such as the Hungarian Method can be used to optimally solve the Assignment Problem, which itself has been shown to have a computational complexity of $O(n^3)$ [48].

In our case blue UAVs are considered agents and are assigned red UAVs to intercept. Originally the author intended to use an objective function in which the cost in matrix C involved solely the Euclidean distances from blue UAV i to red UAV j. Previous work has had UAVs search for stationary targets using this method [13]. However, preliminary tests indicated that this resulted in assignments that were not operationally desirable in protecting the blue team's high value unit (HVU). Reds could get assigned to blues in such a way that the leading blues were assigned to trailing reds as in Figure 2.1.

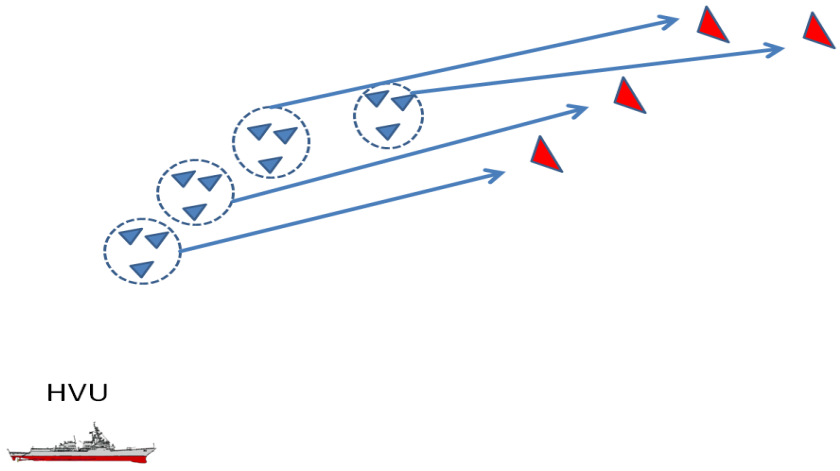


Figure 2.1: Leading red agents assigned to trailing blue groups.[1]

As the goal of the scenario was to protect the blue HVU, this assignment seems counter-productive. The first blue groups fly right past the first oncoming reds in favor of the rear reds. The final blue group must therefore strike with no error. This behavior stems from the assumption of perfect knowledge and zero missed reds during engagement.

It seems wiser to have the front blues engage the front reds and if a red should be missed, there are likely still subsequent blue agents in its path to which it might be assigned in a later iteration of the assignment problem solver. Also, should a blue find its mark and destroy a red,

[1]Destroyer image obtained from Federation of American Scientists at http://www.fas.org/man//dod-101/sys/ship/ddg82-3.gif

the remaining members of its group are available to be assigned to the remaining oncoming reds, resulting in reinforced blue subteams, which increases blue subteams' effective kill probabilities (see Section 3.2.4).

To solve this problem, we considered modifying the objective function to maximize the average distance of any red to the HVU. We also considered an objective function that minimizes the sum of all distances traveled by all red agents. Both of these methods are analogous to our original linear program (which minimizes total distance traveled by blue agents), both still assume perfect knowledge and zero missed reds, and both still suffer from the operationally undesirable assignments demonstrated in Figure 2.1.

Our solution instead involves modifying the cost matrix itself, using the distance of each red from the blue HVU and use that as a discount factor on the entries in the cost matrix. Each entry in the cost matrix, C, of Equation 2.1.1 is multiplied by a discount factor n/j^2, where j is the index of a red agent in a list red agents sorted by their Euclidean distances from the blue HVU. The resulting objective function tends to result in operationally desirable assignments, as in Figure 2.2.

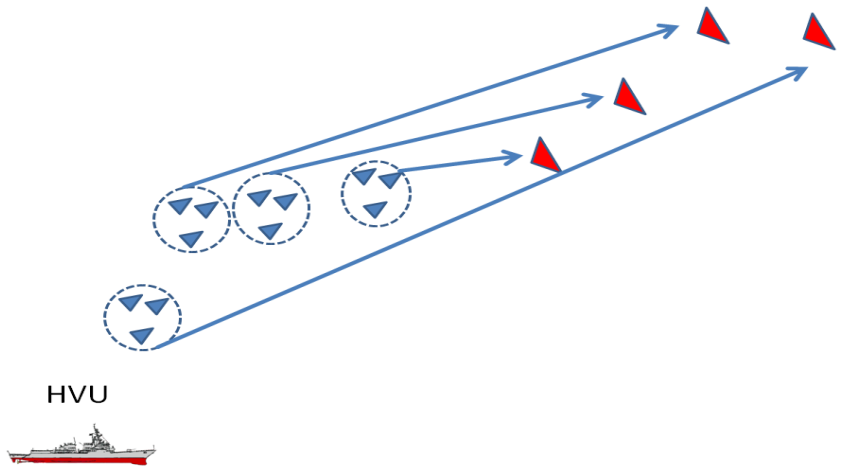


Figure 2.2: Leading red agents assigned to leading blue groups. Note that if leading blue groups successfully engage their targets, the remaining members of those groups are available for reassignment.

The cost matrix is defined only in terms of distances, but the fact that reds and blues are moving should not be discounted, therefore individual red and blue telemetries and velocities are required to obtain predicted positions within some time window. The predicted positions are

then used to generate the distances need to create the cost matrix. However, [49] points out that Euclidean distances do not take into account flyable trajectories (e.g., bounded curvature constraints on fixed wing UAVs), so it is possible that a UAV would be assigned a target it cannot reach. We assume that assignments are to be made at sufficient distances so as to negate this issue, and leave more sophisticated trajectory generation research for future study.

This method also does not account for the proximity of red threats to the blue HVU, only allows a single blue to be assigned a single red (which assumes 0% failure rate for blues and no false negatives from sensor readings), does not take false detections into account, ignores the possibility of new reds arriving to engage the HVU, and does not handle reassignment well when combat causes the number of blues to exceed the number of reds. The issue of new reds arriving can be dealt with immediately by periodically rerunning the algorithm and assuming there are a sufficient number of blues available for launch to handle any number of reds that might arrive. Some of the other issues can be handled by formulating the problem as an instance of the transportation problem, of which the assignment problem is a special case [50, 48].

2.2 Centralized Algorithm B - Redundant Targeting

The transportation problem is typically described in terms of shipping material from sources to sinks; there are *m* sources where material is available and *n* sinks where material is required. There are also vectors **a** and **b**, where a_i denotes the units of material available at source *i*, and similarly b_j denotes the units of material required at sink *j*. There is an $m \times n$ cost matrix C which denotes costs of shipping the unit at $C_{i,j}$ from source *i* to sink *j*.

$$\begin{aligned} \min \quad & \sum_{i=1}^{m} \sum_{j=1}^{n} C_{i,j} X_{i,j} \\ \text{s.t.} \quad & \sum_{j=1}^{n} X_{i,j} = a_i \text{ for } i = 1 \dots m \\ & \sum_{i=1}^{m} X_{i,j} = b_j \text{ for } j = 1 \dots n \\ & X_{i,j} \in \{0, 1\} \text{ for all } i, j \end{aligned} \tag{2.2.1}$$

We apply the formulation given as Equation 2.2.1 to our scenario as follows: sources correspond to m blue UAVs and sinks to n red UAVs. Each $a_i = 1$ so that each blue is only assigned a single red and each $b_j = s$, where s is the size of a squad of blues assigned to a single red –

more than one blue can now be assigned to each red. The authors in [46] call this special case of the transportation problem where $a_i = 1$ the semi-assignment problem and state that it can be optimally solved in $O(n^2m)$ time.

A necessary condition for the feasibility of this linear program is that the data for the problem must satisfy the balance condition:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j \tag{2.2.2}$$

In the case of excess supply or excess demand, simple modifications to the cost matrix and the source and sink vectors before executing the Hungarian Method ensures that the problem is still solvable [48]. Suppose there is a situation where the equation $g = \sum a_i - \sum b_j$ yields a non-zero g. If supply is greater than demand ($g > 0$), a new entry is added to **b**, $b_{n+1} = g$ (representing an imaginary sink demanding the leftovers), and a new column is introduced to C (and n is incremented by 1). Cost is assumed to be 0 to send to that sink so all values in the column are 0, and excess units are assigned to that column. Other the other hand, if cost is greater than supply ($g < 0$), $-g$ new entries are added to **a**, all having value 1 (representing an imaginary sources holding the missing units), and $-g$ rows of zeros are introduced to C (and m is incremented by $-g$).

While this is an elegant mathematical solution to cases of excess supply or demand, it is important to understand its implications. In the case of excess demand, the unacceptable condition of underassigned or unassigned reds would arise. In the case of excess supply, excess blue UAVs would be underutilized. In an operational context, these implications would need to be dealt with by modifying the value of s, applying a priority filter to the red UAVs, or some other workaround.

The presented approach ensures that there is never excess demand by launching blue UAVs as necessary; in other words, ensuring $\sum_{j=1}^{n} b_j \geq ns$. We are therefore making the assumption of an endless supply of blue UAVs. The case of underutilized blue UAVs can be handled by a

slight modification to the linear program formulation:

$$\begin{aligned} \min \quad & \sum_{i=1}^{m}\sum_{j=1}^{n} C_{i,j}X_{i,j} \\ \text{s.t.} \quad & \sum_{j=1}^{n} X_{i,j} = 1 \text{ for } i = 1 \ldots m \\ & \sum_{i=1}^{m} X_{i,j} \geq s \text{ for } j = 1 \ldots n \\ & X_{i,j} \in \{0, 1\} \text{ for all } i, j \end{aligned} \tag{2.2.3}$$

The only change is to the second constraint: it now gives reds a lower bound, *s*, on the number of blues assigned to them, but more blues may be assigned to a red if there are more available. For example, if there are false detections of reds or combat reduces the number of reds, then any blues that were assigned to a now defunct red are assigned to another red with the next iteration of the solver. The second constraint ensures that this occurs only when excess blues are available.

2.3 Decentralized Algorithm A - Implicit Coordination

One decentralized solution that has been explored is to have each agent compute a centralized solution for all agents [32, 33, 34, 36]. Each agent chooses its own optimized solution and proceeds to execute it. In simulation, this approach is possible and agents do not even need to coordinate their decisions if they all have perfect global situational awareness. Each agent comes to the same conclusion as all others as to what the optimal assignments should be and act accordingly.

2.4 Decentralized Algorithm B - Market-Based

An alternative to seeking an optimal solution is to trade optimality for reduced computational complexity. Market-based algorithms using economic principles have been used to assign agents to tasks in previous work [40, 31, 34, 42, 41, 39]. Market-based solutions are dependent on the problem being solved, in this case we have a market in which blues attempt to purchase reds via auction. In each round of the auction blues are each given an equal amount of money and the cost they are willing to pay for each red based on Euclidean distance (blues bid higher for closer reds). We propose Algorithm 1 for conducting an auction to determine assignments.

Algorithm 1 Auction algorithm employed in this study

$m \leftarrow$ number of blues, $n \leftarrow$ number of reds, $s \leftarrow$ preferred size of blue subteams
$agentMatrix \leftarrow$ 2D array of blues agents to reds agents
$totalRedAssignments \leftarrow 0$, $totalBlueAssignments \leftarrow 0$
$bidFactor \leftarrow s(s+1)/2$
for $i = 1 \rightarrow m$ **do**
    sort ith row of the $agentMatrix$ array by euclidean distance
end for
while $totalBlueAssignments < m$ **do**
    for $i = 1 \rightarrow m$ **do**
        $numBidsMade \leftarrow 1$
        for $j = 1 \rightarrow n$ **do**
            if red at $agentMatrix[i][j]$ has $< s$ assignments **or** $totalRedsAssignments >= sm$ **then**
                $nextBid \leftarrow (s - j) \times bidFactor$
                blue i places bid of $nextBid$ on red $agentMatrix[i][j]$
                $numBidsMade \leftarrow numBidsMade + 1$
                if $numBidsMade == s$ **then**
                    break
                end if
            end if
        end for
    end for
    iterate through bids and determine winners for this round
    increment $totalRedAssignments$ and $totalBlueAssignments$
end while

The complexity of this algorithm is $O(m + ksmn)$, where k is the number of times the while loop iterates (or the number of auction rounds that occur before all blues have purchased a red), m is the number of blue agents, n is the number of red, and s is the minimum size of a blue subteam. Note that k is significantly lower than m and n as m grows, and s is, in fact, a constant that does not depend on m or n at all, simplifying the complexity statement to $O(mn)$. Through simulation we investigate whether this algorithm takes both less computation than a centralized algorithm while still hopefully approaching the optimal solution. One of our experimental design factors employs this algorithm with a single broker. Another employs multiple brokers and is discussed in the next section.

2.5 Decentralized Algorithm C - Implicit Market-Based

To this point, we have not addressed the fact that all the previous algorithms assume global situational awareness of all agents. Of course, in more practical settings all agents have limited and imperfect information [35]. We combine the idea of implicit centralized coordination with our market-base solution to address this. Each agent employs the more efficient auction algorithm locally, effectively acting as its own broker. Each agent must use only situational information from its own sensors, and situational data transmitted to it via a simulated wireless network. An arbitrary limit of a 10 nautical mile sensing sphere is imposed upon each blue agent; any contact outside this range is assumed undetected. Sensor noise is also modeled for blue agents' target; sensors get progressively less noisy as a target gets closer.

Communication between agents that are using implicit coordination can help overcome the lack of global situational awareness [32, 37], but care must be taken in creating communication protocols [38]. In our case, UAVs may be temporarily out of range from one another so agents must not expect to send or receive data immediately at all times. Additionally, there are bandwidth constraints on how much data can be shared between individual UAVs, subteams of UAVs, and the entire blue team. We assume the existence of an effective communication network and leave the research of such to future study.

2.6 Decentralized Algorithm D - Fixed Assignment

The simplest form of decentralized assignment would be to perform no assignment at all. To examine whether or not performing centralized and decentralized assignment has a positive effect on the ability of blue agents to stop incoming red agents, this study allows scenarios in which blue agents never change the assignment initially received at launch.

2.7 Assignment Persistence

Recall that we model the cost of assignment based heavily on Euclidean distance. Even with the discount factors mentioned in Section 2.1, as a subteam of blue agents approaches a substeam of red agents the distances between individual reds and blues will get closer and closer, making assignment oscillation likely. In order to reduce unnecessary oscillation between assignment, two factors are introduced in the experimental design: a commit distance and a persistence factor. The commit distance is the distance at which blue is from its target that it ceases updating its assignment (see Figure 3.4). The persistence factor is a discount introduced to the cost of the

red agent to which a blue is already assigned, making it more likely that the blue will keep its current assignment (see 3.2.4).

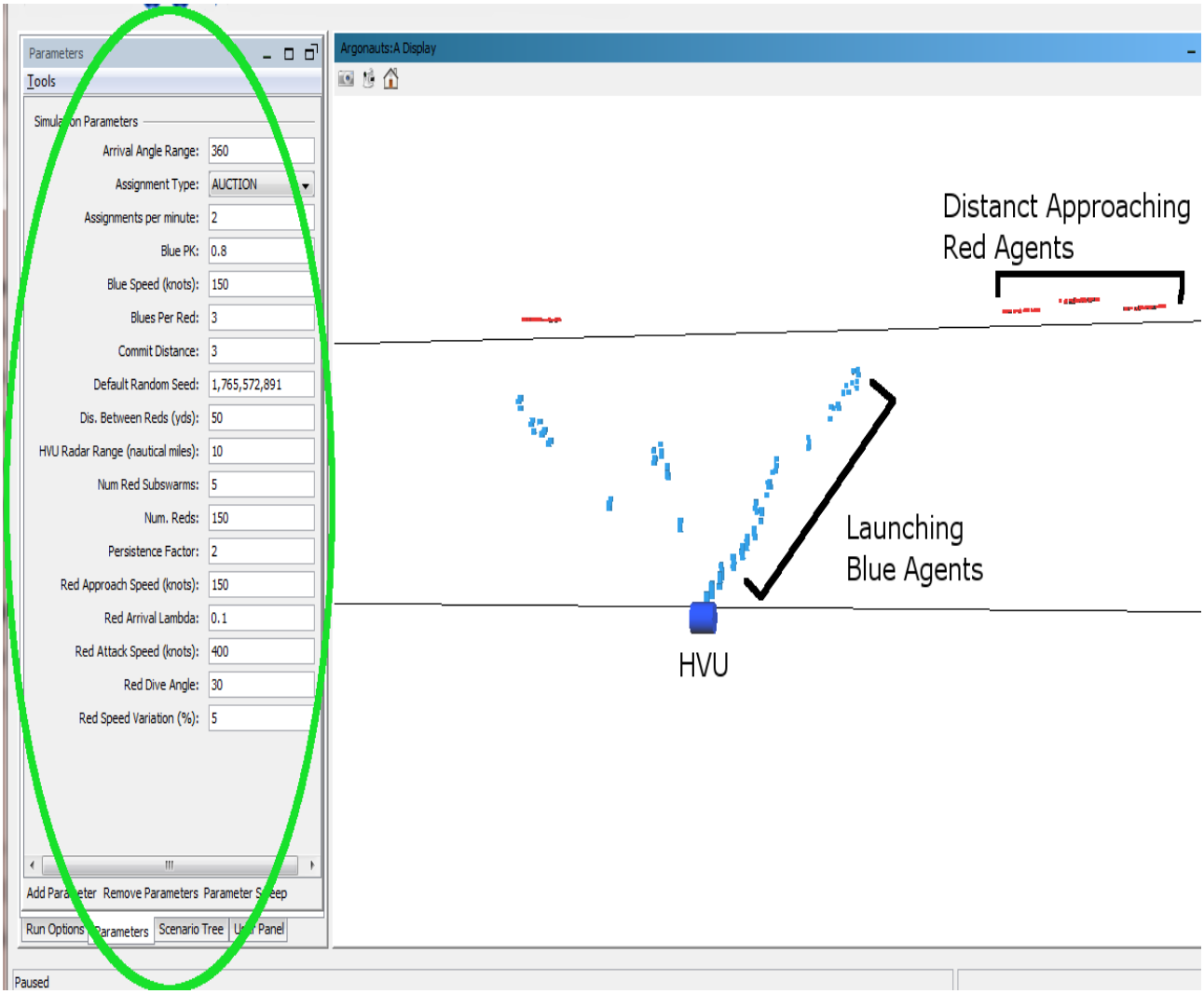


Figure 2.3: Repast Simphony user interface. Parameters section is circled.

2.8 Implementation

From the previously discussed algorithms, we choose to implement four approaches for our simulation.

- A centralized algorithm that solves the transportation problem via an open source linear programming solver called the GNU Linear Programming Kit (GLPK) [51].

- An auction algorithm based on the Algorithm 1 which uses a single broker with global situational awareness.
- An auction algorithm based on Algorithm 1 in which each agent acts as its own broker, using limited situational awareness as described in Section 2.5.
- A control algorithm in which blues never change the assignment received at launch.

A scenario begins with red agents approaching the blue HVU. When a red arrives within a predetermined secure perimeter (see Figure 3.1), s blue agents are launched to counter the threat. As more reds arrive, more blues are launched and assignments are exchanged according to the assignment algorithm chosen for that scenario. Blues pass through three phases as they approach and engage reds.

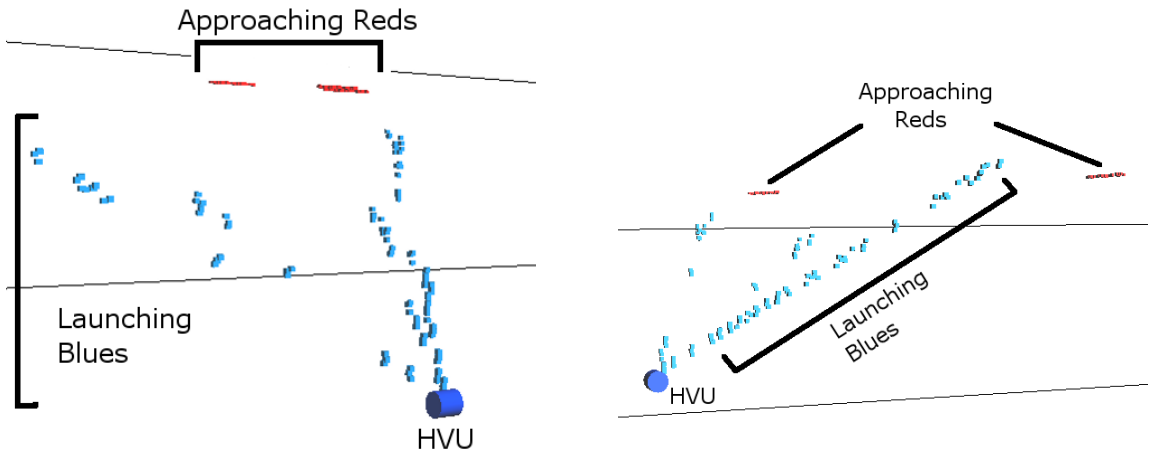
2.8.1 Simulator

The open source agent-based simulator Repast Simphony [43] is used to test each assignment algorithm. This simulator allows agent behavior be defined via the Java programming language. We have created three agent types for this experiment: a blue HVU agent, a blue UAV agent, and a red UAV agent. The blue HVU is capable of launching blue UAV agents to defend itself. The blue UAV agents seek to engage red UAV agents. Red UAV agents fly at high altitude until they reach a desired dive angle (see Section 3.2.12), and then dive to engage the blue HVU. This experiment studies the case where reds attack a single blue HVU.

Repast Simphony allows simulation parameters to be entered before a run (see Figure 2.3). The simulator can be run in visualization or batch mode. In batch mode a different set of parameters can be applied for each simulation batch, facilitating Monte Carlo analysis. Repast parameters map directly to experimental design factors outlined in Chapter 3.

2.8.2 Launch Phase

Blues take off in teams of size s to engage each encroaching red. During the launch phase, blues head in the direction of their target and using an arbitrary angle of climb of 30 degrees (see Figure 2.4). During this phase, no attempt is made to keep the team together via flocking or any other group cohesion mechanism. The high angle of climb and ignoring the team emphasizes the goal of this phase: of reaching the altitude of the target as fast as possible. When engaging the target we wish for the altitude of both the blue and red to be about equal.



(a) (b)

Figure 2.4: Two different views of blue agents in launch phase in the Repast Simphony visualization tool. Distant red agents can also be seen approaching in the background. The blue HVU is the cylinder in the foreground.

At any time during this phase any blue agent may be reassigned to another red, depending on how reds are arriving. In that case blues simply turn towards the new red target and fly as fast as possible towards it with the indicated angle of climb.

Figure 2.5: Agents in Repast Simphony. Blue agents on the left have not yet left launch phase. The other blue agents on the left have entered transit phase. Red agents are on the right side.

2.8.3 Transit Phase

As soon as a blue agent's altitude is greater than or equal to that of its target it leaves the launch phase and enters its transit phase (see Figure 2.5). During this phase Boids flocking [25] is employed to keep blue teams in cohesive spatial units. During this phase a blue agent may be reassigned to another red, in which case it changes teams and begin flocking with its new team members.

The blue agent transits toward the red target as fast as possible while flocking until it reaches a predefined commit distance from its target. At that point, it leaves transit phase.

2.8.4 Engage Phase

Upon leaving transit phase, a blue agent enters engage phase. The blue leaves the pool of blues who are available for reassignment and the assignment algorithm ceases to operate on it. The blue ceases flocking and flies at maximum speed towards its target, attempting to attack it. The blue is able to to estimate the velocity of the red agent in order to fly a course to close within the blue's attack range. An attack does not occur until the blue agent's distance to its target is within the effective range of its weapon. The effective range of blue agents' weapons in this scenario is set to 10 meters based on information obtained from open literature on explosive warheads (see Figure 3.3). Any time a blue fires its weapon it is removed from the scenario because we assume that the weapon is either one-time use or self destructive in nature. There are three possible outcomes of the engagement:

1. The blue is unable to get within effective distance of its weapon and fails to attack the red. Since reds are assumed to not employ any avoidance maneuvering, this only tends to happen if the blue is significantly slower than the red.

2. The blue gets within the effective distance of its weapon and attacks the red but fails to destroy or disable it. The blue is removed from the scenario as a result of the attack, while the red continues toward the HVU undeterred.

3. The blue gets within effective distance of its weapon, attacks the red and destroys both the red and itself.

After a successful attack, any remaining blues assigned to the destroyed red leave the engage phase and re-enter the transit phase, at which point they are assigned to a new red via the assignment algorithm in use for the current simulation run.

CHAPTER 3:
Experiment Design

We employ design of experiments (DOX) techniques to lay out our empirical model. We first perform a series of screening experiments in order to discover which factors are most significant with respect to the response variable. We use a D-optimal design in order to reduce the variability made in the predictions made by our model. After performing a sufficient number of trials, multivariate linear regression is employed to identify significant factors.

3.1 Response Variable: Percentage of Reds Destroyed

This experiment seeks to discover how effective a given assignment algorithm is, subject to a number of constraints or factors. The higher the percentage of reds destroyed, the more effective a particular design point is. The percentage is measured via Monte Carlo runs, which takes factors as input parameters to each simulation run. This percentage is a good indicator of how well the experiment is meeting the primary goal of blue's defence: prevention of reds reaching the blue HVU. We use the percentage of reds destroyed rather than the absolute number of reds destroyed since the latter can be derived from the former – we know how many reds were launched (see Section 3.2.2).

Response variables that were considered but not used included percentage of blues destroyed, survival of the blue HVU,

3.2 Factorial Design

Factorial experimental designs improve the efficiency of an experiment with many factors by reducing the number of runs necessary to examine the interactions between factors and the effects of factors and their interactions on response variable(s) [52]. Each factor of this experiment, the levels chosen, and their expected impacts are discussed in this section. A complete list of this experiment's factors is in Table 3.1.

3.2.1 Assignment Method

This factor is the major focus of this experiment. This is a categorical factor with discrete levels denoted Centralized, Auction, Local Auction, and Fixed, representing the algorithms discussed in Chapter 2. It is hypothesized these cases affect the response variable in decreasing order of effect as follows: (1) Centralized, (2) Auction, (3) Local Auction, and (4) Fixed. Centralized should result in the highest percentage of reds killed since it is optimal, Local Auction should perform worse than Auction because of the loss of perfect global situational awareness. The control case of Fixed should perform the worst, especially in cases where reassignment after blue take off is beneficial, such as reds flying faster than blues, all reds arriving nearly simultaneously, fair number of red subteams (see Section 3.2.13), and high red subteam arrival angle range (see Section 3.2.15).

3.2.2 Number of Reds

This is the total of number of reds to attack the blue HVU during the course of the experiment and varies from 5-150. This factor is not expected to affect the response variable, or the percentage of reds to hit the HVU. However, it is expected that the higher the number of reds, the more likely the red team is to make hits on the HVU, which will be of interest to studies seeking not only to increase the percentage of reds destroyed but also to minimize the total number of hits on the blue HVU.

3.2.3 Secure Perimeter Radius

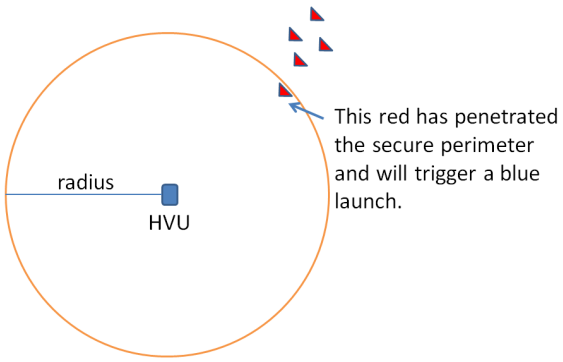


Figure 3.1: Red agent entering the secure perimeter of the blue HVU.

This is the radius of the protected airspace around the blue HVU. Any red arriving inside the

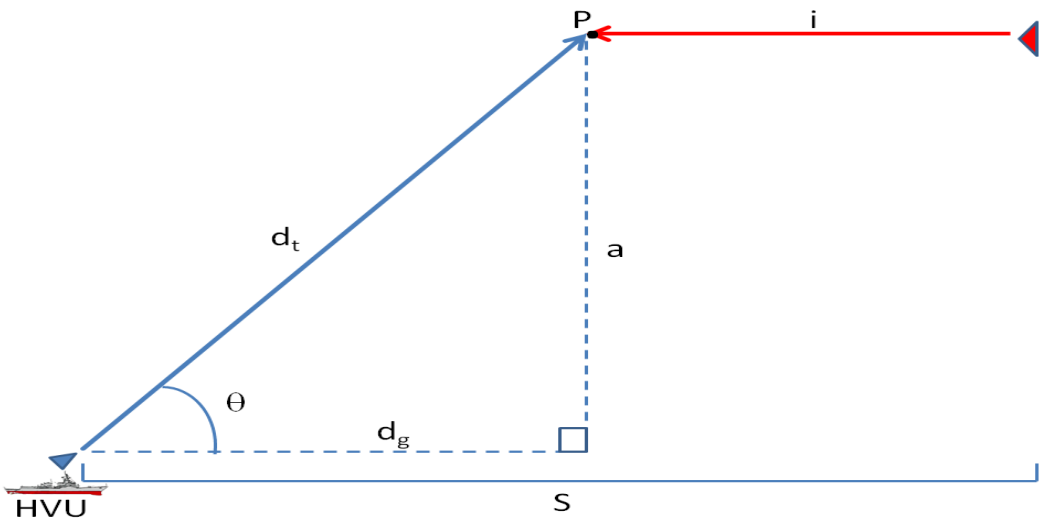


Figure 3.2: A red agent approaching the blue HVU. The blue agent must take off when the red is at a sufficient distant, S (or the Secure Perimeter Radius), so that the blue will reach point P before the red does. The values of the other variables are discussed in the text.

perimeter triggers a launch of a blue subteam to engage the red (see Figure 3.1). It is expected that this perimeter will have a lower bound below which the percentage of reds killed will drop dramatically because the blues will be unable engage the reds before they begin diving (see Section 3.2.12). We chose 10 nautical miles as our worst case as we expect to have blue agents with speeds of at least 80 knots (see Section 3.2.6), and reds with a maximum transit speed of 160 knots (see Section 3.2.9). Our current worst case dive angle is 60° and red's cruising altitude is fixed at 2 nautical miles (12,154 feet) for this experiment. The cruising altitude of current systems [20] are close to 10,000 feet but we expect that innovations in the near future could push that ceiling higher and red will probably desire a high altitude approach to evade ground fire as long as possible in land approaches.

Examining Figure 3.2, given that we know a red's velocity, v_r, a blue's velocity, v_b, and a blue agent's angle of climb, θ, and the altitude at which a red is cruising, a, then it is possible to define minimum the Secure Perimeter Radius, S, at which a blue must take off in order to engage red. We assume that if a red agent passes point P before blue is able to arrive there and transition out of its Launch Phase then it will be very difficult for blues to turn and engage their targets before reds begin their dives. Red must ingress a distance of i to reach point P, and blue must cover d_t as it takes off (simultaneously covering an overground distance of d_g). We can formalize what the minimum Secure Perimeter Radius, S, must be as follows:

$$
\begin{aligned}
d_t &= \frac{a}{\sin\theta} \\
d_g &= d_t \cos\theta \\
i &= S - d_g \\
&\text{if blue is to reach point P before red:} \\
S &> d_t(\frac{v_b}{v_r} + \cos\theta)
\end{aligned}
\tag{3.2.1}
$$

In the case where red has the highest advantage in speed over blue (reds at 160 knots and blues at 80), the security perimeter must be above 5.46 nautical miles or blues will be unable to engage reds at all. Since we have chosen 10 nautical miles as our security perimeter, we do not expect this to be a significant factor.

3.2.4 Number of Blues per Red

As each red arrives inside the secure perimeter defined above, a subteam of blues launches to attempt to engage it. This factor is the number of blues in each blue subteam upon launch from the HVU. The following equation models what the effective probability of kill is on a given red that is being attacked by s blues:

$$P_k^{eff} = 1 - (1 - P_k^{blue})^s \tag{3.2.2}$$

where P_k^{blue} is the probability of successful kill for a single blue against a red, and P_k^{eff} is the probability of successful kill of a subteam of blues of size s. Therefore, as the number of blues per red, s, increases the percentage of reds killed increases. When P_k^{blue} is 0.85, P_k^{eff} passes 0.99 at three blues per red. Thus, we choose three as the maximum number of blues in the experiment. A discussion of why we believe 0.85 is a good value for PK_b follows.

3.2.5 Blue P_k

This is the probability of a successful kill when a single blue attacks a red. Attacks occur when a blue agent is within 10 meters of its red target based on warheads in the missiles outlined in Figure 3.3 and we expect that present day warheads are at least as capable as those presented [11, 16, 17, 18]. Blues are assumed to either be destroyed in the engagement or employ a non-resuable weapon so they are removed from the simulation after engagement.

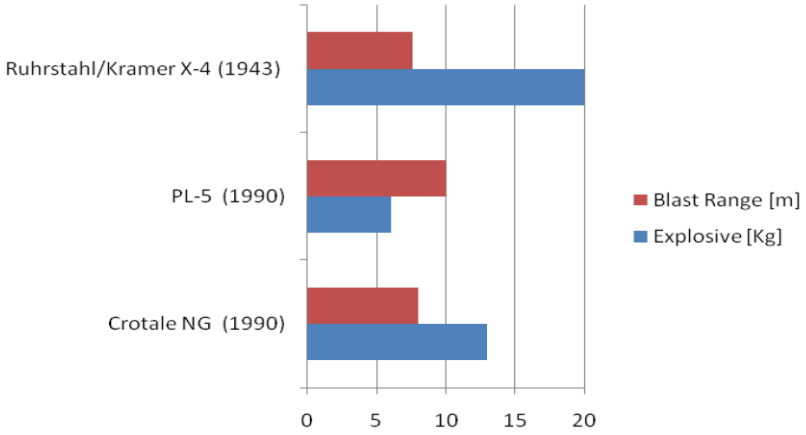


Figure 3.3: Blast range of warheads of representative anti-air missiles from open literature.

.

A valid P_k for an explosive air-to-air projectile is exceedingly difficult to find in open literature. As [53] states, "this type of data is inevitably classified." However, using available open data for AA cannon systems [19], we are able to arrive at an estimate of 0.85. In the experiment we vary this factor between 0.6 and 0.95 in order to account for different possible weapons systems. Substantially lower P_k values are left to future study. It is expected that as P_k increases the percentage of reds killed increases.

3.2.6 Blue Max Speed

This is the maximum speed of blue agents in knots. Note that in this simulation there is a fixed 10% variance for each blue's maximum speed to account for possible environmental disturbances such as wind and turbulence. This factor is likely to interact with the red transit speed factor since it is expected that as blue gains a higher advantage in speed over red that the percentage of reds destroyed increases in this manner:

$$\beta(v_b - v_r) = \% \text{ reds destroyed} \tag{3.2.3}$$

where β is a constant and v_b is blue max speed factor and v_r is the red max speed factor. Ideally, blue agents would at least match red agents' speed in an engagement.

A survey of current systems that could be outfitted for the scenario we propose yields maximum speeds of 90-135 knots [20, 21, 22, 23]. We have a lower level of 80 knots to allow for systems

not moving at top speeds, and and upper level of 160 knots to anticipate future technologies.

3.2.7 Commit Range

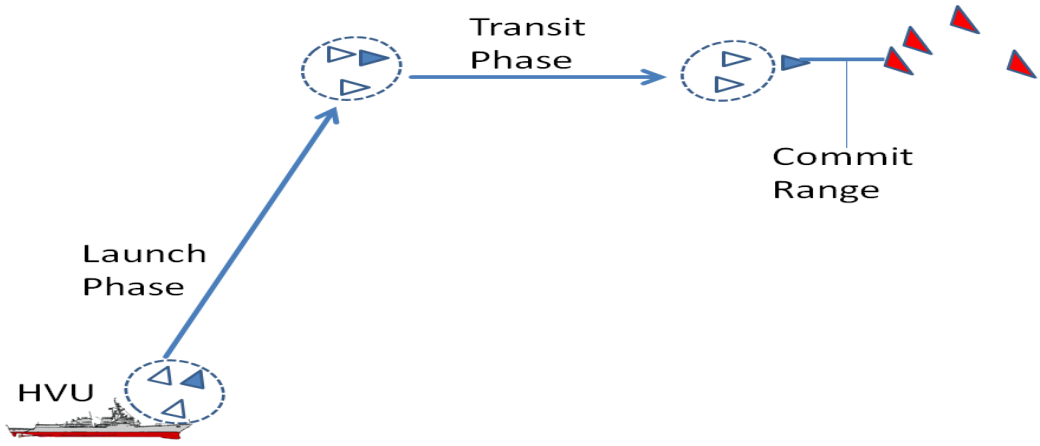


Figure 3.4: Lifetime of a blue agent. The end of the Transit Phase is determined by the Commit Range factor. Once a blue enters within that distance of its red target it ceases to update its assignment, stops flocking with its subteam, and enters Engage Phase.

Commit range is the distance at which a blue leaves transit phase (see Figure 3.4), commits to its current target, stops flocking with its subteam, and ceases updating its assignment, i.e. the blue is locked onto its assigned target. This factor and the persistence factor were introduced to reduce assignment oscillation. As a subteam of blues approaches a subteam of reds the Euclidean distances between all agents begin to become indistinguishable and therefore less useful for assignment. Thus, it is more likely that reassignments will be made.

It is expected that as this factor increases it will decrease the amount of assignment oscillation. There should be a lower bound on this factor below which there is too much oscillation in the system and there will be a corresponding low percentage of reds destroyed. There will also be an upper bound where the blue agents commit far too soon and never take advantage of reassignments when they should. Above this upper bound the percentage of reds killed is expected to be low. This experiment therefore seeks to discover what those upper and lower bounds are and also predict what value might maximize the response variable.

3.2.8 Persistence Factor

This factor exists to introduce inertia into the assignment algorithms. We make it less likely for a blue to leave its current assignment during a reassignment cycle by introducing a discount for

the cost of the red the blue is currently assigned to. If a blue agent is already assigned to a red agent then the cost of assignment to that red agent is determined by:

$$c_{\text{new}} = c_{\text{original}}/p \tag{3.2.4}$$

where p is the persistence factor. This is a weighting factor applied in each assignment algorithm, where assignment cost of a red agent is applied as in Equation 3.2.4 if a blue is already assigned to that red. As p increases, the probability of maintaining an assignment increases. As with commit range, this factor likely has a lower bound below which there is too much oscillation between assignments and an upper bound above which not enough reassignments occur. We seek to predict what value maximizes the percentage of reds killed. Levels for this experiment are set at 1 as the low and 25 as the high.

3.2.9 Red Transit Speed

This is the speed (in knots) that reds move towards the blue HVU. It is likely there is an interaction between red and blue transit speeds in affecting the response variable since it is probable that the faster reds travel with respect to blues, the lower the percentage of reds killed with be (see Section 3.2.6). Levels for this factor are the same as for those in Section 3.2.6, 80 knots as the low value, and 160 knots as the high.

3.2.10 Red Speed Variation

A red may be assigned a speed within this percentage of the red transit speed. This factor is introduced due to the expectation that there is some variability in the efficiency of red agents' engines and other control hardware that affect flight dynamics since we assume a lack of prior knowledge about the precise speed of the adversary. For clarity, red transit speed is affected by red speed variation as follows:

$$s_{\text{final}} = s_{\text{nominal}} \pm \frac{v}{2}(s_{\text{nominal}}) \tag{3.2.5}$$

where v is red speed variation, s_{nominal} is red transit speed, and s_{final} is the red agent's transit speed after red speed variation has been applied.

This factor impacts how tightly packed red subteams remain as they approach. It is expected that this factor should not have a noticeable effect on percentage of reds destroyed due to the

assumption that blues have the ability to estimate red velocities (see Section 2.8.4). Levels for this experiment are set to 0% for low and 20% for high.

3.2.11 Red Max Dive Speed

This is the terminal velocity of a red as it dives towards the blue HVU. Since we assume red dive velocities are so much greater than blues are able to employ during these dives, it is expected that once a red has started a dive it will be nearly impossible to stop it and so the author does not expect this factor to have much effect on percentage of reds destroyed. This factor, however, may be relevant for other layered defense elements that may be employed in future studies. This reinforces a goal of this experiment to engage and destroy reds before they start their dives. Levels in this experiment are set to 300 knots at the lower level and 400 knots at the upper.

3.2.12 Dive Angle

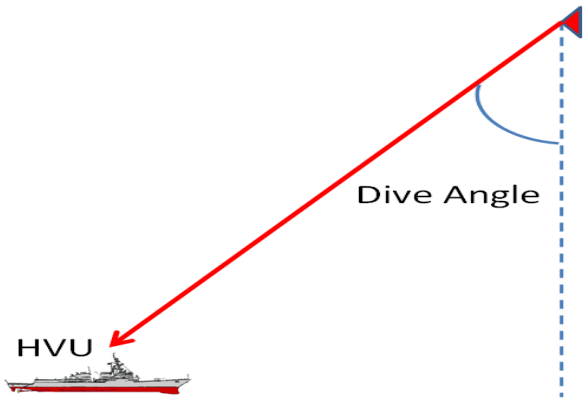


Figure 3.5: Illustration of how dive angle is defined.

Fig 3.5 illustrates a dive angle. When a red reaches that angle, it begins its dive. The larger this angle is, the sooner the red team can begin their dive. The simulator does not model flight dynamics, so the speed of the red is assumed to be fixed during its descent. The author expects that as dive angle increases, the percentage of reds destroyed will decrease. The low level for this factor is 5 °, and the high level is 60 °.

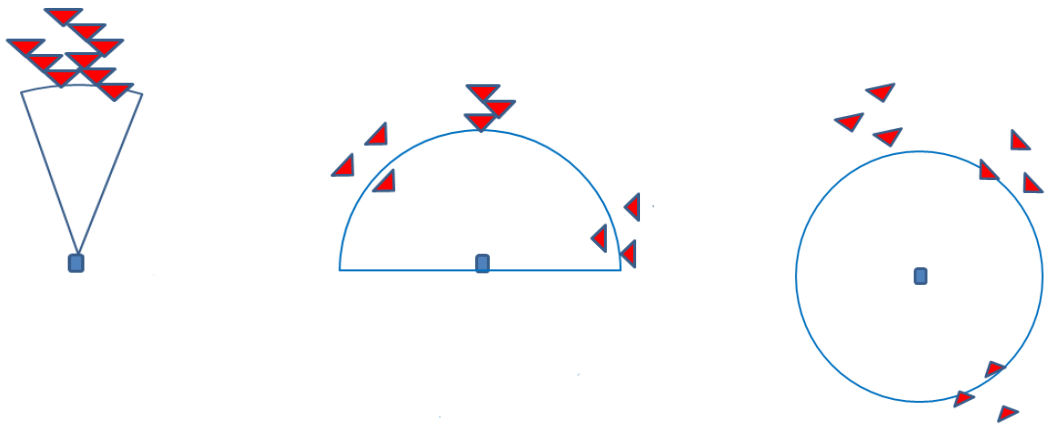
3.2.13 Number of Red Subteams

The red team is be divided into a number of subteams. Each subteam arrives according to the red subteam arrival λ parameter. It is expected that this value will interact with the red subteam arrival range parameter. As both number of subteams and arrival range increase the percentage of reds destroyed should decrease. Levels for this factor are 1 for the low and 150 for the high. In cases where the number of subteams is greater than the number of reds the number of subteams is clamped to the number of reds.

3.2.14 Red Subteam Arrival λ

Red subteams arrive according to a Poisson Process. Poisson process interarrival times are exponentially distributed and can be completely defined with a single term, λ [24]. For this factor, λ represents arrivals per second; note that as λ increases interarrival times decrease. It is expected that as λ increases the percentage of reds destroyed will decrease as more reds will be present and decrease the amount of blue agents that can be reused after an engagement. The low level for this factor is 0.001 arrivals per second and the high level factor is 1000000 arrivals per second (high level chosen to model simultaneous arrival of all red subteams).

3.2.15 Red Subteam Arrival Angle Range



(a) (b) (c)

Figure 3.6: Various arrival angle ranges: (a) a 10 ° arrival angle range, (b) a 180 ° arrival angle range, (c) a 360 ° arrival angle range.

We define a circular region around the blue HVU. This factor determines what sector of that circle any given red subteam may arrive from (see Figure 3.6). The factor varies from a small 10 °

sector to the complete 360° sector. The former case represents reds arriving from essentially a single launch point while the latter represents possible launch points completely surrounding the HVU. It is expected that as this factor increases the percentage of reds destroyed will decrease, due to the splitting of blue forces and mitigating the effect of blue reinforcements.

3.2.16 Initial Distance Between Reds

When a red subteam arrives this factor determines the distance each red agent is from another. Together with the red speed variation factor, this factor determines the tightness of a red subteam's formation. It is expected that this parameter will not have a significant effect on the percentage reds destroyed, although it may play a role in realistic settings, e.g., target identification and tracking in cluttered environments. This factor has as low level of 10 yards and a high level of 500 yards.

3.3 Factor Screening

Factor screening is the process of systematically varying input factors in order to identify factors that significantly affect the response variable [54]. We employ the JMP® 9 Pro statistical software package [55] to assist in this process. JMP has several DOX tools to identify the number of design points necessary, examine interactions between factors, and perform linear regression. Screening experiments often are conducted using two-level factorial designs, a low level and a high level, generally -1 and 1. Center points can also be added to verify linearity within the model and check for variance at the center of the design space (center points therefore take a coded value close to zero). The factor levels in Table 3.1 must be therefore be coded to fit within -1 and 1. This equation was used on all factors:

$$coded = \frac{value - (Factor_{low} + Factor_{high})/2}{(Factor_{high} - Factor_{low})/2} \tag{3.3.1}$$

except the Red Subteam Arrival λ factor. Since the maximum value was so high we did not wish to use a center point at 50000, as it would reset in nearly identical interarrival times. Instead this formula was used for that factor:

$$coded = \frac{\ln(value) - (\ln(Factor_{low}) + \ln(Factor_{high}))/2}{(\ln(Factor_{high}) - \ln(Factor_{low}))/2}. \tag{3.3.2}$$

Consider the two-level factorial model to which we wish to fit data:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n + \beta_{12} x_1 x_2 + ... + \beta_{(n-1)n} x_{n-1} x_n + \epsilon \tag{3.3.3}$$

where y is the response variable, each x_i is a factor, each $x_i x_{i+1}$ is a level two interaction, each β_i is coefficient that denotes the effect of a factor or factor interaction upon the response, β_0 is the intercept, and ϵ is the error term. All of the runs in an experimental design can be written as a system of equations using the coded values in Section 3.3 for the factors as follows:

$$\begin{aligned}
a_1 &= \beta_0 + \beta_1(-1) + \beta_2(-1) + ... + \beta_n(-1) + \beta_{12}(-1)(-1) + ... + \beta_{(n-1)n}(-1)(-1) + \epsilon_1 \\
a_2 &= \beta_0 + \beta_1(1) + \beta_2(-1) + ... + \beta_n(-1) + \beta_{12}(1)(-1) + ... + \beta_{(n-1)n}(-1)(-1) + \epsilon_2 \\
a_3 &= \beta_0 + \beta_1(-1) + \beta_2(1) + ... + \beta_n(-1) + \beta_{12}(-1)(1) + ... + \beta_{(n-1)n}(-1)(-1) + \epsilon_3 \\
a_4 &= \beta_0 + \beta_1(-1) + \beta_2(-1) + ... + \beta_n(1) + \beta_{12}(-1)(-1) + ... + \beta_{(n-1)n}(-1)(1) + \epsilon_4 \\
&... \\
a_f &= \beta_0 + \beta_1(1) + \beta_2(1) + ... + \beta_n(1) + \beta_{12}(1)(1) + ... + \beta_{(n-1)n}(1)(1) + \epsilon_f
\end{aligned} \tag{3.3.4}$$

where a_i is here defined as the i^{th} value of the response variable and f is the sum of all factors and factor interactions we wish to model. This is more concisely expressed in matrix form:

$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where

$$\mathbf{y} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ \vdots \\ a_f \end{bmatrix}, X = \begin{bmatrix} 1 & -1 & -1 & \cdots & -1 & 1 & \cdots & 1 \\ 1 & 1 & -1 & \cdots & -1 & -1 & \cdots & 1 \\ 1 & -1 & 1 & \cdots & -1 & -1 & \cdots & 1 \\ 1 & -1 & -1 & \cdots & 1 & 1 & \cdots & -1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \cdots & 1 & 1 & \cdots & 1 \end{bmatrix}, \tag{3.3.5}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{12} \\ \vdots \\ \beta_{(n-1)n} \end{bmatrix}, \text{and } \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \vdots \\ \epsilon_f \end{bmatrix}$$

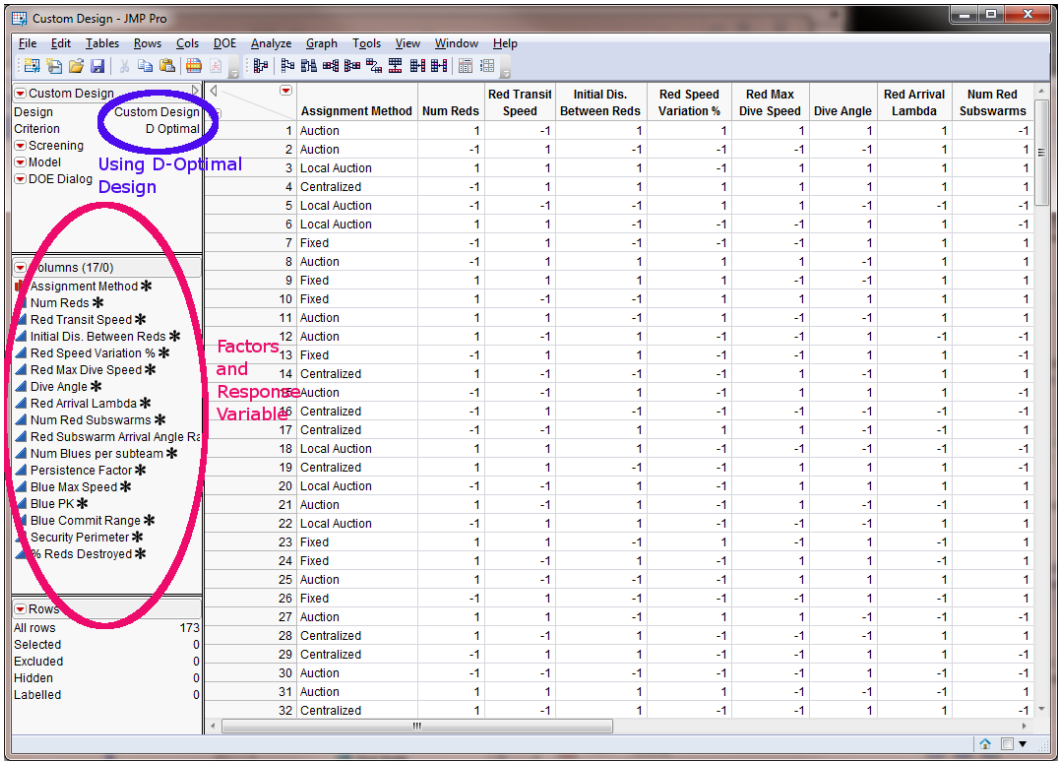
The least squares estimates of the model coefficients are the values of the β's that minimize the sum of squares of the model errors. These are determined by:

$$\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T \mathbf{y} \tag{3.3.6}$$

A design that minimizes the variance of the model regression coefficients is called a D-optimal design and the D terminology is used because the design is found by selecting runs to maximize the determinant of $X^T X$ [52]. JMP provides a tool that generates design points corresponding to a given experimental design (see Figure 3.7). The design matrix generated for this experiment resulted in 173 design points including 4 center points. The center points are added to check for curvature in the regression.

3.3.1 Asymptotic Variance

A number of design points were examined in order to decide on the number of iterations the simulator would perform for each design point, see Figure 3.8. We ran the simulator for these design points many times until the standard deviation of the response variable stabilized. Sev-



	Assignment Method	Num Reds	Red Transit Speed	Initial Dis. Between Reds	Red Speed Variation %	Red Max Dive Speed	Dive Angle	Red Arrival Lambda	Num Red Subswarms
1	Auction	1	-1	1	1	1	1	1	-1
2	Auction	-1	1	-1	-1	-1	-1	1	1
3	Local Auction	1	1	1	-1	1	1	1	1
4	Centralized	-1	1	1	1	1	1	1	1
5	Local Auction	-1	-1	-1	1	1	-1	-1	-1
6	Local Auction	1	1	-1	-1	-1	1	1	-1
7	Fixed	-1	1	-1	-1	-1	1	1	1
8	Auction	-1	1	1	1	1	-1	1	1
9	Fixed	1	1	1	1	-1	-1	1	1
10	Fixed	1	-1	-1	1	1	1	1	1
11	Auction	1	1	-1	1	-1	-1	1	1
12	Auction	1	-1	1	-1	-1	1	-1	-1
13	Fixed	-1	1	1	-1	-1	1	1	-1
14	Centralized	1	-1	-1	-1	1	-1	1	1
15	Auction	-1	-1	1	-1	1	1	1	-1
16	Centralized	-1	1	-1	-1	-1	-1	-1	-1
17	Centralized	-1	-1	1	1	1	-1	-1	1
18	Local Auction	1	1	1	-1	-1	-1	-1	-1
19	Centralized	1	1	-1	-1	1	1	1	-1
20	Local Auction	-1	-1	1	-1	1	1	1	1
21	Auction	1	-1	1	-1	1	-1	-1	1
22	Local Auction	-1	1	1	-1	-1	-1	1	1
23	Fixed	1	1	-1	1	-1	-1	1	1
24	Fixed	1	-1	1	-1	1	1	-1	1
25	Auction	1	-1	-1	-1	1	1	1	1
26	Fixed	-1	1	-1	-1	-1	1	-1	1
27	Auction	1	1	-1	1	1	-1	-1	-1
28	Centralized	1	-1	1	-1	-1	-1	1	1
29	Centralized	-1	1	1	-1	-1	1	1	-1
30	Auction	-1	-1	-1	-1	-1	1	-1	-1
31	Auction	1	1	1	1	-1	-1	-1	1
32	Centralized	1	-1	1	-1	-1	1	1	-1

Figure 3.7: JMP screen for designing a D-Optimal experiment.

enty five simulation runs was deemed sufficient to obtain stability. With 173 design points this resulted in 12975 simulation runs. A single run lasts minutes to six hours depending on the values of the design points.

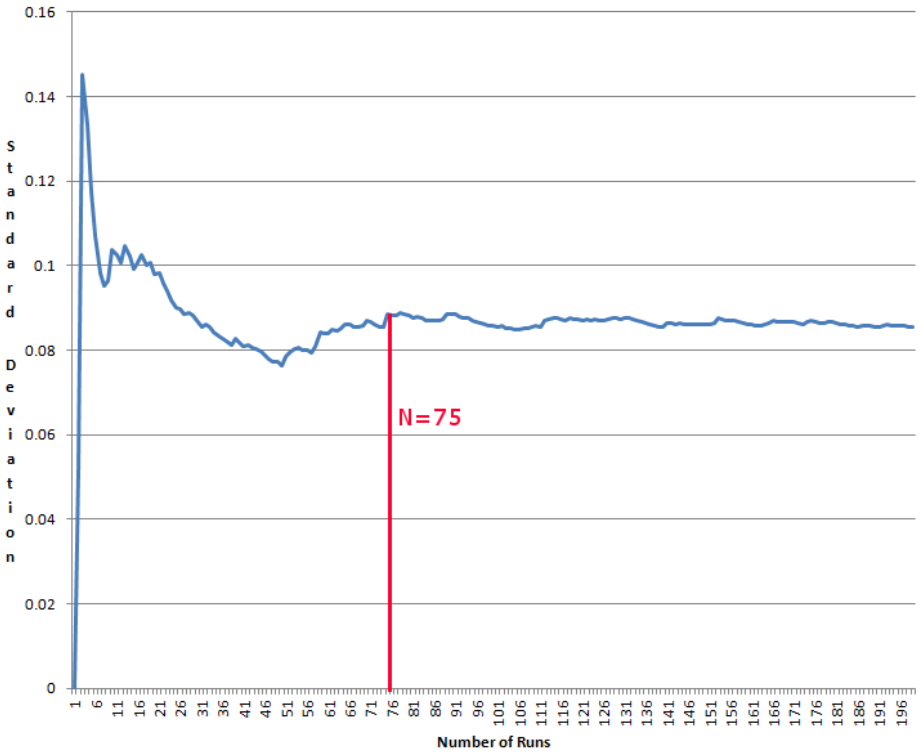


Figure 3.8: Asymptotic Standard Deviation for a single design point.

Factor	Type	Levels	Description
Assignment Method	Categorical	Centralized	Algorithm used to assign blue agents to red agents.
		Auction	
		Local Auction	
		Fixed	
Number of Reds	Continuous	5, 77, 150	Size of Red Team
Number of Blues per Red	Continuous	1, 2, 3	Size of blue subteams
Blue P_k	Continuous	0.6, 0.775, 0.95	Probability of kill for a single blue attacking a single red.
Blue Max Speed	Continuous	80, 120, 160	Maximum speed of blue agents (knots).
Commit Range	Continuous	1, 3, 5	Range at which blues cease updating their assignment (nautical miles).
Persistence Factor	Continuous	1, 13, 25	Determines how costly it is to change a blue agent's current assignment.
Secure Perimeter Radius	Continuous	10, 30, 50	Distance a red must be from the HVU before blue subteams are launched against it (nautical miles).
Red Transit Speed	Continuous	80, 120, 160	Speed reds move towards HVU (knots).
Red Speed Variation	Continuous	0%, 10%, 20%	Transit speed varies by this percentage.
Red Max Dive Speed	Continuous	300, 350, 400	Dive speed (knots).
Dive Angle	Continuous	5, 32.5, 60	Angle at which reds dive toward HVU (in degrees).
Number of Red Subteams	Continuous	1, 30, 150	Red Team is broken into subteams. This is how many subteams arrive.
Red Subteam Arrival λ	Continuous	0.001, 0.1, 1000000	Rate at which red subteams arrive (arrivals per second).
Red Subteam Arrival Angle Range	Continuous	10, 185, 360	Determines where red subteams arrive in relation to the HVU (degrees).
Initial Distance Between Reds	Continuous	10, 255, 500	Distance between individual red agents in a subswarm upon arrival (yds).

Table 3.1: Experiment Design Factors

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Analysis

We here present the results of the screening experiments designed in Chapter 3 and identify significant factors and factor interactions found during experimentation. Detailed analysis is provided for factors found to be the most significant including number of blues per red, blue P_k, red transit speed, and number of reds. A predictive model using the most significant factors is presented, as are simulation test runs to validate the model. The most unexpected result is that assignment method does not significantly impact the response variable.

4.1 Logistic Regression

Our response variable is bounded by 0 and 1, representing a percentage of reds that were destroyed, and this fact must be considered while fitting a prediction model with respect to our factors. A standard least squares approach can lead to values that are outside the range of our response variable. The logistic function is useful here in that it allows us to transform our data into a range that is between $-\infty$ and ∞ in order to fit our model. Once the model has been chosen the predicted values can be transformed back into percentages. The logit function is defined by [52] as:

$$y^\star = \ln\left(\frac{y}{1-y}\right), \tag{4.1.1}$$

where y is our response variable and $y^\star$ is the logit of the response variable. To transform the response variable back to a percentage after regression, we use the inverse logit:

$$y = \frac{1}{1+e^{-y^\star}}\,. \tag{4.1.2}$$

Care must be taken when our response variable takes on the value of zero or one as $\ln(0)$ is undefined, and we also cannot divide by zero so a response value of one is unusable. Our results contain no response values of zero, but some did have the value of one. To cover that case we subtracted all values by 0.001 before transformation and added that value back after transformation.

4.2 Significant Factors

A standard least squares regression (via Equation 3.3.6) of the data reveals that we can fit the data to a line with an adjusted R^2 of about 0.98, (see Figure 4.1). However, this fitting results in a prediction model incorporating 50 factors and interactions with p-values less than 0.05.

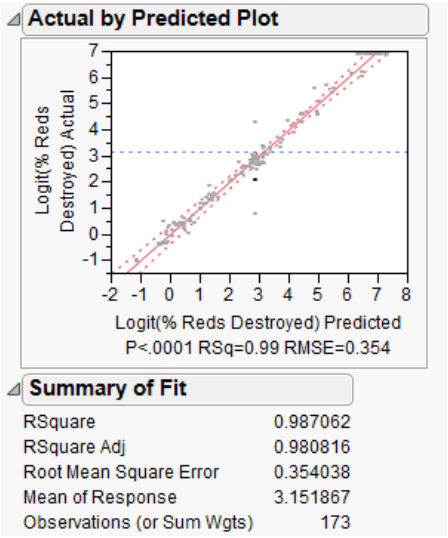


Figure 4.1: Least squares regression of data with the maximum adjusted R^2 calculated via JMP.

Relaxing adjusted R^2 to 0.90 results in a fit that is not so tight (see Figure 4.2), but results in fifteen factors and interactions, shown in Figure 4.3. Significant factors are ("$\times$" is used to denoted interactions between factors):

- Number of Blues per Subteam
- Blue P_k
- Red Transit Speed
- Blue Max Speed
- Number of Reds $\times$ Red Transit Speed
- Red Transit Speed $\times$ Number of Blues per Subteam
- Number of Reds
- Number of Red Subteams $\times$ Number of Blues per Subteam

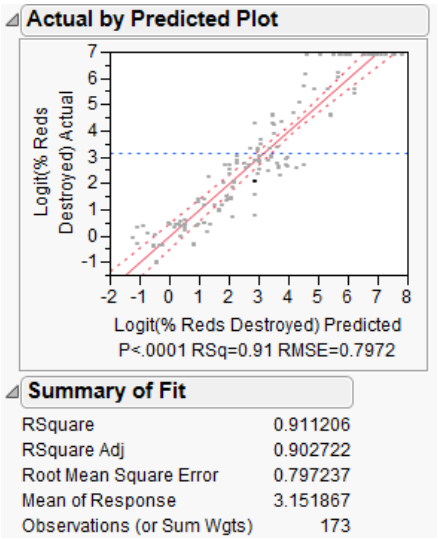


Figure 4.2: Least squares regression of data with adjusted R^2 of 0.90. Calculated and plotted via JMP.

- Red Transit Speed $\times$ Blue Max Speed
- Number of Reds $\times$ Initial Distance Between Reds
- Red Dive Angle $\times$ Blue Max Speed
- Red Subteam Arrival Angle Ranges $\times$ Blue P_k
- Assignment Method (when set to Local Auction) $\times$ Blue Max Speed
- Assignment Method (when set to Auction) $\times$ Blue Max Speed
- Assignment Method (when set to Centralized) $\times$ Blue Max Speed

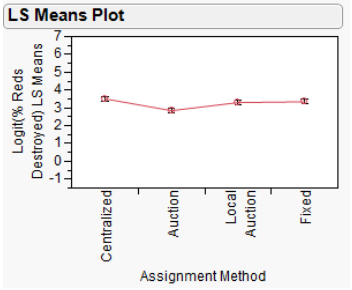
What is immediately evident is that assignment method by itself is not identified as one of these top fifteen factors. It only appears as a part of a mixed interaction with blue max speed, and factors other than assignment method have a much greater impact on the response variable. Examination of stepwise fitting to shows that these three interactions taken together account for less than one percent of the variability in the model (see Figure 4.5). Although assignment is not significant we still wish to discover if market based solutions approach the centralized solution in their effect on the response variable.

We are able to verify our assumption that auction and local auction assignment methods approach the optimality of the centralized solution by examining the least squares mean of the

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Num Blues per subteam	1.8036517	0.066098	27.29	<.0001*
Blue PK	1.2007463	0.066574	18.04	<.0001*
Red Transit Speed	-0.823716	0.066746	-12.34	<.0001*
Blue Max Speed	0.6579804	0.06489	10.14	<.0001*
Num Reds*Red Transit Speed	-0.465235	0.066652	-6.98	<.0001*
Num Red Subswarms*Num Blues per subteam	-0.365973	0.06587	-5.56	<.0001*
Red Transit Speed*Num Blues per subteam	-0.363497	0.066811	-5.44	<.0001*
Num Reds	-0.329948	0.063227	-5.22	<.0001*
Dive Angle*Blue Max Speed	-0.254907	0.067477	-3.78	0.0002*
Red Transit Speed*Blue Max Speed	0.2511004	0.067603	3.71	0.0003*
Red Subswarm Arrival Angle Range*Blue PK	0.2341694	0.067478	3.47	0.0007*
Num Reds*Initial Dis. Between Reds	0.2100193	0.06881	3.05	0.0027*
Assignment Method[Local Auction]*Blue Max Speed	0.3401222	0.113562	3.00	0.0032*
Assignment Method[Auction]*Blue Max Speed	0.1367793	0.12013	1.14	0.2566
Assignment Method[Centralized]*Blue Max Speed	-0.003661	0.107713	-0.03	0.9729

Figure 4.3: Significant factors identified in order of significance in regression with adjusted R^2 of 0.90



	Logit	Percentage
Centralized	3.5123798	0.971037967
Auction	2.8478546	0.945207678
Local Auctio	3.3474359	0.966020771
Fixed	3.303272	0.964540889

Figure 4.4: A least squares mean study showing that different assignment methods do not vary greatly in how they affect the response variable.

response when comparing only assignment methods (see Figure 4.4). However, the fixed assignment approaches the centralized as well! It appears that the other factors in this scenario render assignment relatively insignificant and highlights the complexity and variability present in such operational contexts.

4.2.1 Residuals

It is necessary to verify that the least squares regression used to create a factorial model adequately models the error in Equation 3.3.3. This can be done by examining the residuals, or the differences between predicted values and observed values for all data points. All residuals need to be normally distributed, independent, uncorrelated, and have a mean close to zero.

Since response variable values were obtained by unrelated runs of a simulator, each having a different random seed, the residuals are independent by construction. To show the residuals are

Step History

Step	Parameter	Action	"Sig Prob"	Seq SS	RSquare	Cp	p
1	Num Blues per subteam	Entered	0.0000	481.8777	0.4288	48238	2
2	Blue PK	Entered	0.0000	243.0305	0.6450	29913	3
3	Red Transit Speed	Entered	0.0000	88.81125	0.7241	23218	4
4	Blue Max Speed	Entered	0.0000	83.87355	0.7987	16895	5
5	Red Transit Speed*Blue Max Speed	Entered	0.0001	20.42014	0.8169	15357	6
6	Num Reds*Initial Dis. Between Reds	Entered	0.0002	16.48957	0.8316	14116	7
7	Num Red Subswarms*Num Blues per subteam	Entered	0.0002	14.9202	0.8448	12993	8
8	Num Reds*Red Transit Speed	Entered	0.0001	15.61548	0.8587	11817	9
9	Red Transit Speed*Num Blues per subteam	Entered	0.0000	15.32467	0.8724	10664	10
10	Num Reds	Entered	0.0000	17.6632	0.8881	9333.7	11
11	Dive Angle*Blue Max Speed	Entered	0.0010	8.243041	0.8954	8714.1	12
12	Red Subswarm Arrival Angle Range*Blue PK	Entered	0.0019	6.872473	0.9015	8197.9	13
13	Assignment Method{Auction&Local Auction&Fixed-Centralized}*Blue Max Speed	Entered	0.0010	10.87474	0.9112	7383.8	16
14	Num Blues per subteam*Blue Commit Range	Entered	0.0032	5.43164	0.9160	6976.2	17

Figure 4.5: Steps in the stepwise regression listed in JMP. Assignment method enters the model as a factor on step 13 and changes R^2 from 0.9015 to 0.9112, thus accounting for only 0.0097 of the variability in the model.

approximately normally distributed we can do a fitting of the residuals to a normal distribution (see Figure 4.6). This same procedure shows the mean of the residuals is very close to zero.

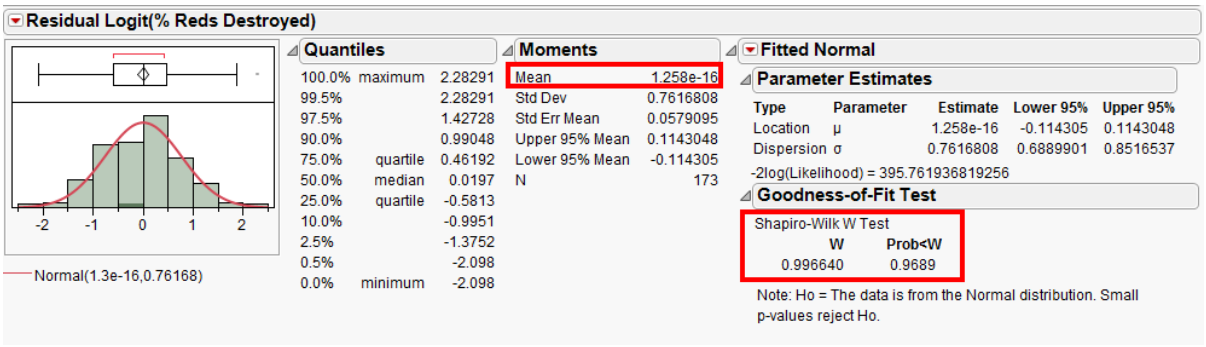


Figure 4.6: Fitting the residuals of the logit of the response variable reveals a near normal fit with mean close to zero.

To demonstrate that residuals are uncorrelated we present a plot of residuals in the order that each experiment was performed in Figure 4.7. There is no pattern in the plot and this is sufficient to demonstrate a lack of correlation among individual residuals.

4.3 Prediction Model

Before presenting the prediction model, it is important to state that a lack-of-fit test revealed that there was curvature in our regression (see Figure 4.8) – a fit test with probability of $F < 0.0001$ is a good indicator of the presence of curvature in the fit. We added 74 data points to our experiment design to include second order quadratics for each factor and redid the fit, but the issue

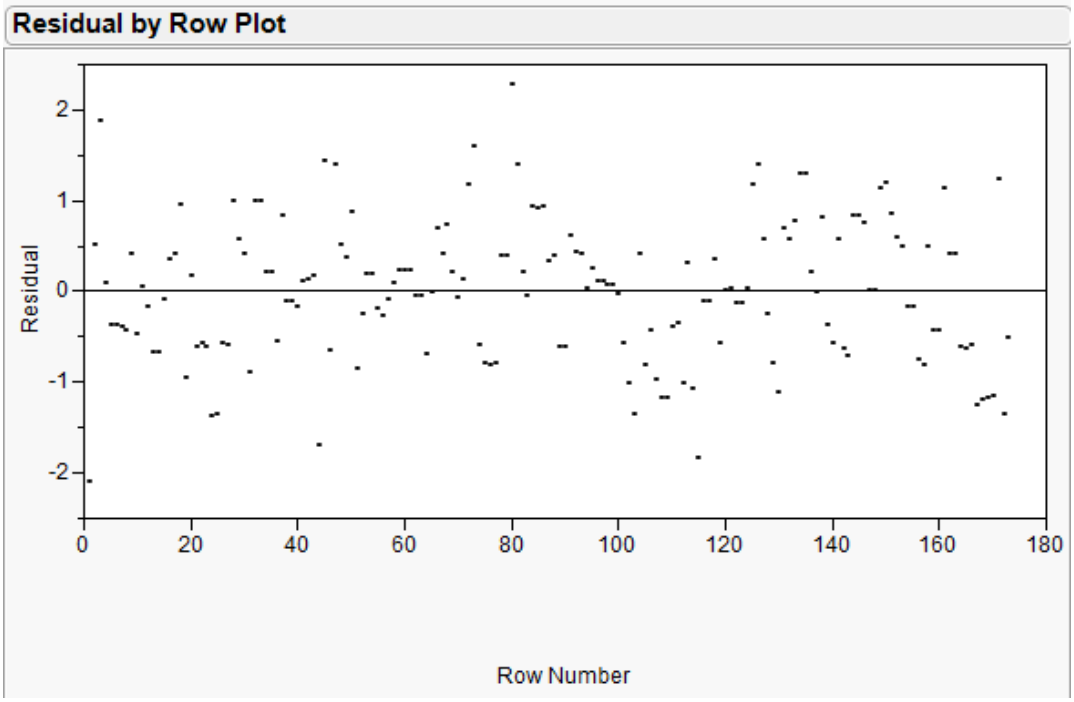


Figure 4.7: No pattern perceived in a plot of residuals of the response variable.

remained. We expect that there is some higher order interaction among one or more the factors, however, we later show in Section 4.3.1 our predictive model is sufficient to conservatively infer outcomes so we have elected to leave the discovery of where these interactions occur to future study. We choose to use the first order predictive model as it is somewhat simpler, yields results comparable to the quadratic model, and still provides meaningful insights into the scenario.

Lack Of Fit

Source	DF	Sum of Squares	Mean Square	F Ratio
Lack Of Fit	93	98.951665	1.06400	81.5084
Pure Error	64	0.835445	0.01305	Prob > F
Total Error	157	99.787109		<.0001*
				Max RSq
				0.9993

Figure 4.8: Lack-of-fit test reveals there is curvature in our fit that we have not accounted for.

Armed with the intercept and coefficients provided for the significant factors in Figure 4.9, we are able to provide the coefficients from Equation 3.3.3 and formalize a predictive model. We choose to limit the predictive model to the top twelve factors since the three assignment factors have a very small effect on the response as well as a higher degree of error . The model therefore is:

$$
\begin{aligned}
y^\star = \; & 2.8505 + 1.8037x_1 + 1.2007x_2 - 0.8237x_3 + 0.6580x_4 - 0.3300x_5 \\
& - 0.4652x_3x_5 - 0.3660x_1x_6 - 0.3635x_1x_3 - 0.2549x_4x_7 \\
& + 0.2511x_3x_4 + 0.2342x_2x_8 + 0.2100x_5x_9
\end{aligned} \tag{4.3.1}
$$

where x_1 is the number of blues per subteam, x_2 is blue P_k, x_3 is red transit speed, x_4 is blue max speed, x_5 is the number of reds, x_6 is the number of red subteams, x_7 is the red dive angle, x_8 is the red subteam arrival angle range, x_9 is the initial distance between reds agents, and $y^\star$ is the logit of the response as described in Section 4.1. To transform $y^\star$ to a percentage we use the inverse logit function, previously presented as Equation 4.1.2. More detailed and specific analyses on individual factors and interactions is in in Section 4.4.

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	2.8504778	0.065225	43.70	<.0001*
Num Reds	-0.329948	0.063227	-5.22	<.0001*
Red Transit Speed	-0.823716	0.066746	-12.34	<.0001*
Num Blues per subteam	1.8036517	0.066098	27.29	<.0001*
Blue Max Speed	0.6579804	0.06489	10.14	<.0001*
Blue PK	1.2007463	0.066574	18.04	<.0001*
Num Reds*Red Transit Speed	-0.465235	0.066652	-6.98	<.0001*
Num Reds*Initial Dis. Between Reds	0.2100193	0.06881	3.05	0.0027*
Red Transit Speed*Num Blues per subteam	-0.363497	0.066811	-5.44	<.0001*
Red Transit Speed*Blue Max Speed	0.2511004	0.067603	3.71	0.0003*
Dive Angle*Blue Max Speed	-0.254907	0.067477	-3.78	0.0002*
Num Red Subswarms*Num Blues per subteam	-0.365973	0.06587	-5.56	<.0001*
Red Subswarm Arrival Angle Range*Blue PK	0.2341694	0.067478	3.47	0.0007*

Figure 4.9: Top twelve parameter estimates (plus intercept) for model fitting provided by JMP.

4.3.1 Verifying the Predictive Model

Test #	# Blues per Red	Blue P_k	Red Transit Speed	Blue Speed	# Reds	# Red Sub-teams	Dive Angle	Arrival Angle Range	Initial Distance Between Reds
1	4	0.92	150	110	40	10	10	180	10
2	1	0.88	100	107	56	4	10	75	50
3	2	0.68	150	90	72	12	30	90	100
4	2	0.60	155	60	36	6	30	270	20
5	1	0.56	120	109	44	4	5	360	50
6	1	0.48	140	90	100	10	30	110	110

Table 4.1: Factor levels for the test design points used to verify the predictive model. All other factors were kept at a constant level.

Six new design points which were not used in the experiment design were chosen and the predictive model was applied to them to verify that it is accurate. Design points were chosen so as to predict 99%, 80%, 75%, 50%, 25%, and 10% reds destroyed. Levels chosen for each factor of each test point are shown in Table 4.1. Each point was run through 75 iterations in the simulator. We found that the predictive model was underestimating the percentage of reds destroyed, and that the gap between the predicted and simulated increased as the predicted value decreased (see Figure 4.10). As it is better to under-predict the number of reds destroyed rather than over-predict we believe this model is acceptable.

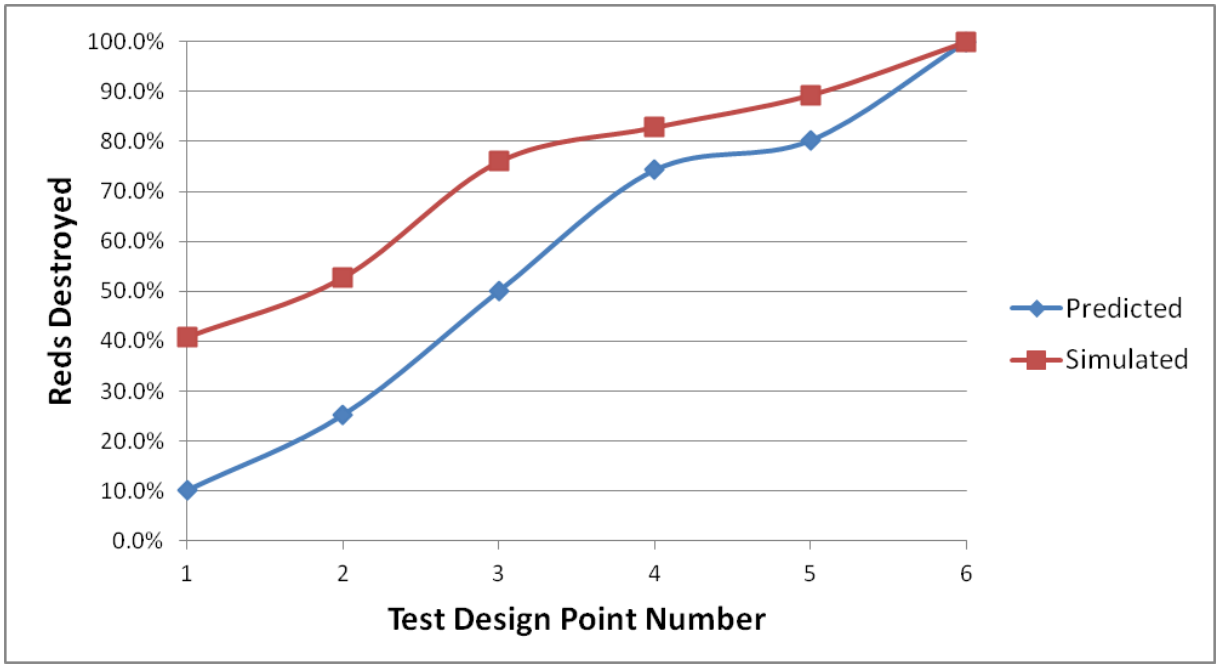


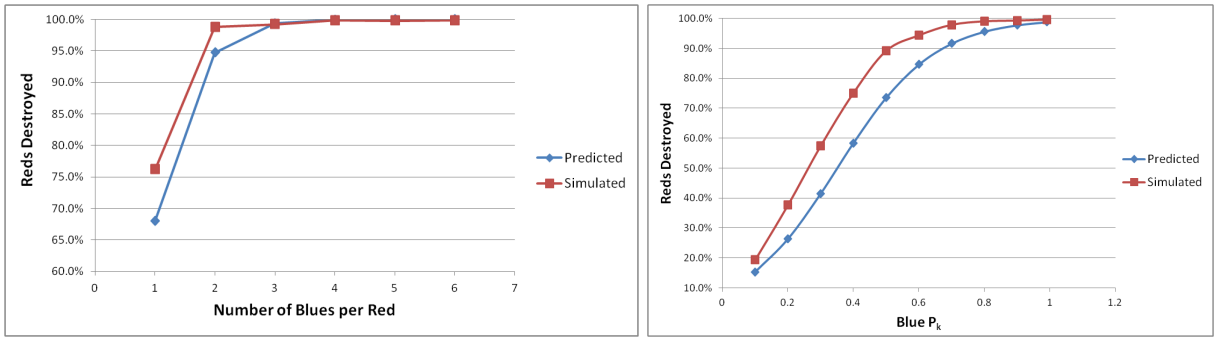
Figure 4.10: Results of tests of the predictive model with the six design points of Table 4.1.

Further analysis was performed to see if it could be discovered which parameters were causing the response to be underestimated by the prediction model. Runs were performed where all factors were held constant save one, i.e one-factor-at-a-time. The factors so examined were:

1. Number of Blues Per Red
2. Blue P_k
3. Red Transit Speed
4. Blue Speed

5. Number of Reds

6. Initial Distance Between Reds



(a) (b)

Figure 4.11: Predictions and simulations of percentage of reds destroyed when varying (a) number of blues per red, and (b) blue P_k.

When examined independent of other factors, the prediction model does well for the number of blues per red and the blue P_k factors (see Figure 4.11). This is likely due to their large effect on the response variable (see Sections 4.4.3 and 4.4.4), and hence their strong role in the prediction model (i.e., they are not strongly influenced by other factors).

Factors blue max speed and red transit speed are less significant than either blue P_k or number of blues per red (see Section 4.4.5 for in depth analysis of these factors significance), and consequently the prediction model appears to overestimate much more when considering these factors independent of all others. However, other factors that contribute even less to the total variability of the regression (based on R^2), such as number of reds and initial distance between reds, suffer from an increased over-prediction as shown in Figure 4.12.

A note should be made here that when blue speed was set to very low, overestimation rather than an underestimation occurs in the predictive model for the only time in our observations. This is because the when blues have a low enough velocity they are unable to reach their red targets' altitudes before the reds begin their dive. This enables reds to strike without being killed by blues at all (see Section 3.2.3).

4.4 Individual Factor Analysis

All significant factors and interactions were evaluated based on results from the regression performed in Section 4.2. Analysis of individual factors is here presented.

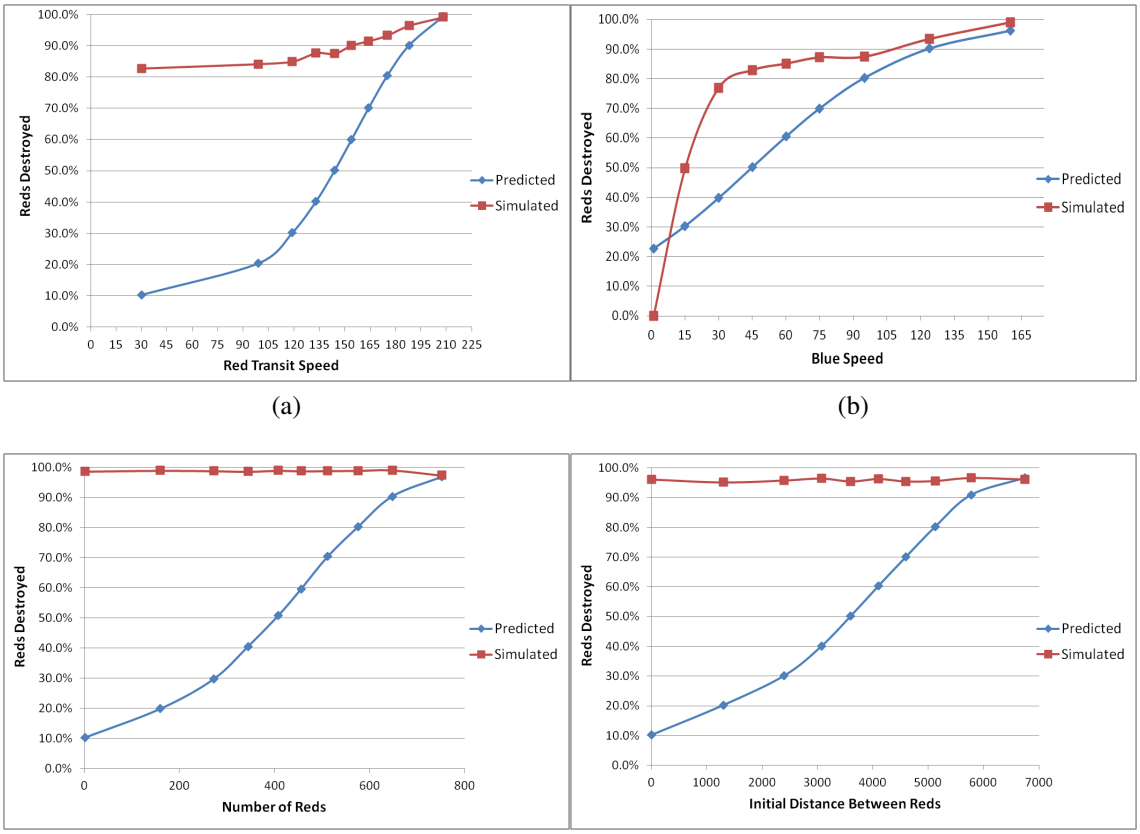


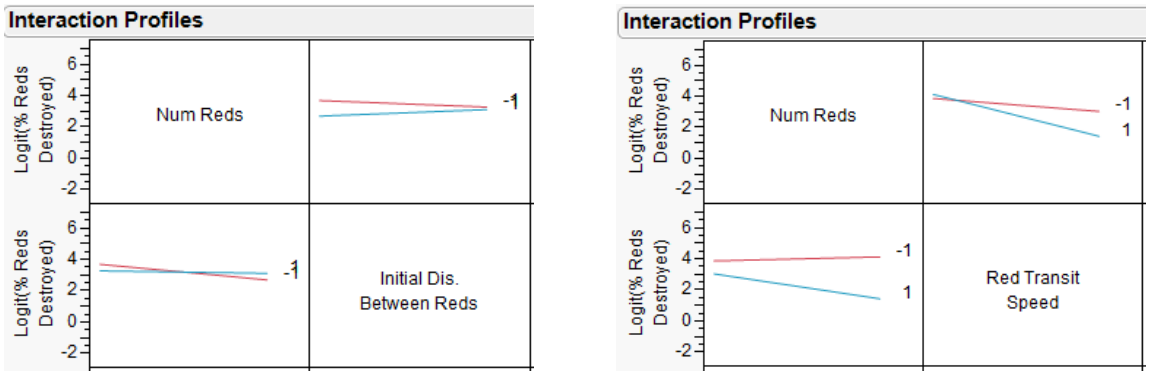
Figure 4.12: Predictions and simulations of percentage of reds destroyed when varying (a) red speed, (b) blue speed, (c) number of reds, and (d) initial distance between reds.

4.4.1 Assignment Method

The main finding of this study is that assignment method is not a significant factor in the percentage of reds destroyed. This may be due to some high order interactions not accounted for in the regression. The fidelity of the simulator should also be considered; radio communications were not modeled in high fidelity, nor were flight dynamics. It would be useful to examine what role assignment may play in other experiments with different constraints, e.g., when the number of blues available for launch is limited.

4.4.2 Number of Reds

We expected that the number of reds would not have a significant effect on the response variable. In fact, as the number of reds increases there is some effect on the response variable, accounting for 1.57% of the variability in the model. Mixed effects in which the number of reds is a factor

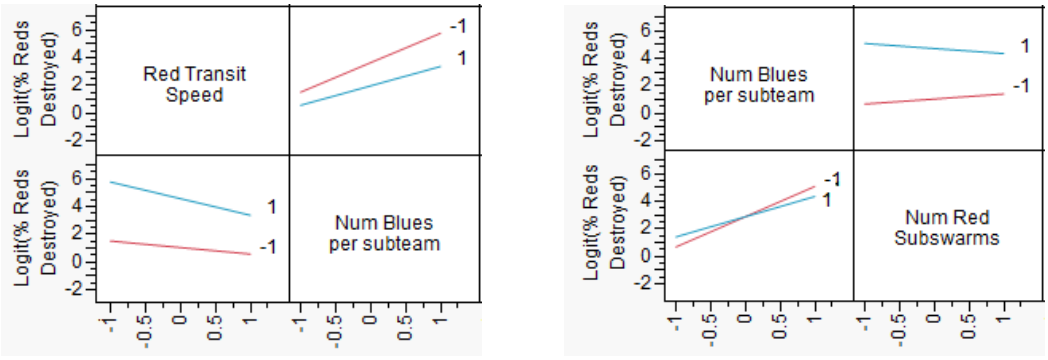


(a) (b)

Figure 4.13: Interactions of the number of reds factor with two other factors: (a) initial distance between reds does interact, but has only a slight effect, while (b) red transit speed interacts much more. Effects are discussed in greater detail in the text.

account for 2.86% of the variability (see Figure 4.5). Noting the interaction of number of reds with the initial distance between reds we believe that tightly packed reds are harder for blues to hit since blues would necessarily be required to fly more tightly packed as they approach, and blues may have more difficulty closing on their targets due to collision avoidance. The interaction with red transit speed has the most effect on the number of reds (see Figure 4.13). Faster reds are simply more effective than slower reds and as more and more fast reds are added the response variable is negatively affected, i.e. fewer reds are destroyed.

4.4.3 Number of blues per red



(a) (b)

Figure 4.14: Interactions of the number of blues per red factor with two other factors: (a) red transit speed has a noticeable effect on the effect blues per red has on the response variable and (b) number of red subteams also has an effect.

We expected that the number of blues per red would positively affect the response variable, and this was shown to be the case. In fact, this factor has the single most significant effect on the response, accounting for 42.88% of the variability in the prediction model. This is not surprising, considering how an increased number of blues per red increases the effective probability of kill of a blue subteam (see Section 3.2.4) as well as providing a store of reinforcement blues as reds are destroyed.

This factor also occurs in two interactions, once with red transit speed, and again with the number of red subteams (see Figure 4.14). In the interaction with red transit speed, as red transit speed increases, the number of blues per red factor still has a positive effect on response, but the effect is slightly diminished: faster reds are better able to escape blues. In the interaction with the number of red subteams, as the number of red subteams increases, the number of blues per red factor has a less positive effect on the response. As the number of red subteams increase they are likely to be spread out spatially and temporally. Blues surviving after a red is destroyed are not as effectively reused because they may simply be unable to reach the remaining reds.

4.4.4 Blue P_k

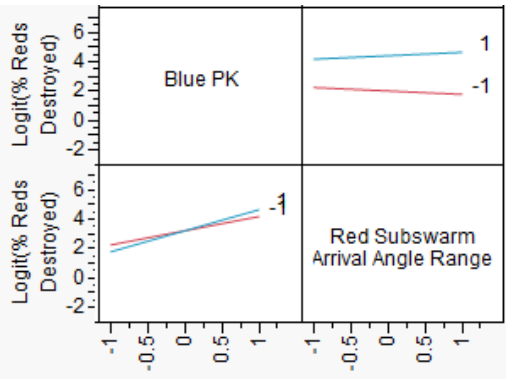


Figure 4.15: Interaction of Blue P_k with red subswarm arrival angle range.

As we expected, blue P_k does positively affect the response variable. In addition, this is the second most significant factor, accounting for 21.62% of the variability in the prediction model (see Figure 4.5). This is intuitive: blues armed with better weapons kill a higher percentage of red. Blue P_k also appears in an interaction with red subswarm arrival angle range (see Figure 4.15). The interaction is very slight, it only accounts for 0.0061% of variability in the model (see Figure 4.5). It is possible that red subteam arrival angle range affects blue P_k, but this is highly unlikely.

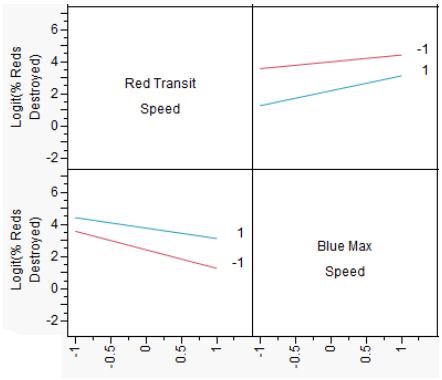


Figure 4.16: Interaction between red and blue speeds.

4.4.5 Blue Max Speed and Red Transit Speed

A significant mixed effect is that between blue max speed and red transit speed (see Figure 4.16). Taken together, blue max speed, red transit speed, and their interaction account for 9.28% of the variability in the prediction model (see Figure 4.5). This is intuitive; as blues' speeds increase while reds decrease, blues should be more effective at closing on their targets.

Red transit speed also appears in an interaction with number of reds (see Section 4.4.2), and number of blues per subteam (see Section 4.4.3). Blue max speed appears in an interaction with dive angle (see Figure 4.17). This was unexpected because we did not expect dive angle to be significant in any way. Blues with low speeds cause dive angle to have a more negative effect on the response. We assume this is due to low fidelity flight dynamics in the simulator. Reds should really dive at a lower velocity if they have a high dive angle. As the simulator currently stands, reds diving at a shallow angle of 60° dive just as quickly as reds diving at a very steep 5°. Reds diving at shallower angles enter their dives sooner and obtain a large speed advantage over blue at that point. However, dive angle should not be completely discounted; even with improved modeling of flight dynamics reds who enter a dive would gain a speed advantage if blues are not diving with them (e.g., blues are engaging from below).

4.4.6 Insignificant Factors

We expected the secure perimeter radius to have an insignificant effect at 10 nautical miles and above in our scenario. This is indeed the case. Both red speed variation and red max dive speed were not expected to have significant effects, nor did they.

Neither commit range nor persistence factor have a significant effect on the response variable. This is not surprising since they are a part of the assignment methods (except Fixed) and assign-

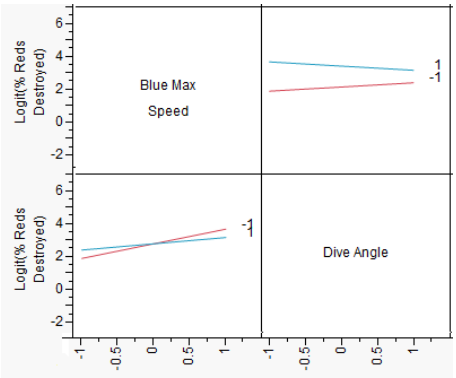


Figure 4.17: Interaction between blue max speed and dive angle.

ment method was also found to not be significant compared to other factors in this experiment.

Red subteam arrival λ is not significant. We had expected it to have a negative effect on the response variable. It is probable that our assumption of always being able to send at least the desired number of blues per subteam to attack any incoming red rendered this factor insignificant.

CHAPTER 5:
Conclusions and Recommendations

5.1 Conclusions

This thesis seeks to determine the effectiveness of various task assignment methodologies for a team of UAVs seeking to thwart an attack by another team of aggressor UAVs. Due to the classified nature of current combat systems data on physical systems that might be employed in such a scenario is unavailable. However, a reasonable simulation model is constructed using data available from open literature. Future studies having access to other data should be able to use specifications from more modern systems without significant modification to the simulated model presented in this study.

Statistical design of experiments and regression modeling is employed as methods of analyses. Using these analyses, we evaluate the degree to which we meet the goals of the study, as defined in Chapter 1:

1. Determine if decentralized assignment methods approach the optimality of centralized methods in complex scenarios with large numbers of agents.

2. Compare assignment methods to other factors in these scenarios and determine how assignment method compares in terms of the effect on the percentage of reds destroyed.

3. Examine the tradeoffs of global vs. local situational awareness vs. performance.

5.1.1 Assignment Methods in Complex Scenarios

Results of this study indicate that factors other than assignment method are far more significant in terms of the effect on the percentage of reds destroyed. It is possible that this is due to a lack of constraints in our scenario design, e.g., an unlimited number of blue agents are available to assign to the incoming red agents. It is also possible that assignment method is, in fact, a less significant determinant for the efficacy of the blue team. Section 5.2 gives recommendations as to how future study might verify the results of this work.

5.1.2 Other Factors' Significance

Factors including number of blues per red, blue P_k, and speeds of blue and red agents proved far more significant than assignment method. These results are rather intuitive, though the author had expected assignment to play a significant role as well.

Number of blues per red proves the most significant, accounting for 42.88% of the variability in the regression. The predictability of the model depends therefore depends heavily on the number of blues per red; the more blues that are available to attack a single red, the higher the blue team's effectiveness. This verifies our hypothesis in Section 3.2.4. This also leads to our conjecture that assignment method may play a greater role in a study where the number of blues is not unlimited; the blue team would have to make careful use of available resources in order to maximize the number of blues assigned to incoming reds without running short on blues for later reds, potentially increasing the impact of assignment method in the blue team's effectiveness.

Blue P_k is the next most significant factor, accounting for 21.62% of the variability in the regression. A more capable blue weapon system is simply more likely to be effective and therefore leads to a more capable blue team overall. However, the number of blues available is a better predictor of the blue team's effectiveness. When faced with a decision to increase the number of blues vice improving individual blue weapon systems0, increasing the number of blues will improve the blue team's performance more quickly, all other things being equal.

Blue and red agent speeds account for 9.28% of the variability in the regression. As blues get faster and reds get slower the blues are more likely to intercept the reds. As reds get faster and blues get slower, blues are less likely to intercept reds. A blue's ability to intercept its red target determines whether or not it is able to engage it at all; if a blue agent is unable to engage its target then the blue agent has no chance of destroying it.

5.1.3 Global vs. Local Situational Awareness

Our results show that there is little difference between the Local Auction (which has local situational awareness) and other assignment methods. It is important to note that our communications were modeled in low fidelity – near perfect communication was assumed. We assumed that including simulated sensor noise in the simulator would be sufficient to differentiate between local and global situational awareness. However, in a scenario in which communications is modeled in higher fidelity in the simulator (or a scenario performed in a physical environ-

ment) the level of situational awareness could affect the blue's performance. The complexities of the model merit further study; including examining higher fidelity communication models.

5.2 Recommendations

An unlimited supply of blue UAVs is not realistic in an operational context and it is possible that this assumption renders assignment method insignificant. Studies which modify both the centralized and decentralized method to incorporate a limited number of available blues may discover that assignment becomes more significant with that additional constraint. It is also unexpected that neither red arrival rate nor red subteam arrival angle is significant. This may be because of a lack of constraints on blues. Increasing the fidelity of the simulator by improving flight dynamics and inducing realistic communication losses or delays may also cause these factors to become more significant.

Field experimentation on physical UAV systems which measure the degree to which assignment method and other factors in this experiment affect mission objectives would also be instructive. Continuing research efforts are underway to perform such live fly field tests at Camp Roberts California.

5.3 Future Work

Information sharing becomes most critical between team members in the engagement phase and segmenting responsibilities for different kinds of information might impact performance. The decision as to which agent is going to engage which target now depends on the combined sensor data of the team. Work in which blues in a subteam have more roles than simply attacker may be of use. For example, some blues could be used to improve distributed sensing on the intended target and others for battle damage assessment.

It is possible that flocking is unnecessary in order for blue teams to complete their missions effectively. Since reassignment occurs throughout the scenario, blues are changing teams as they need to and it is probable in this scenario that flocking does not affect the response variable at all. Indeed, flocking negatively impacts an individual blue's velocity towards its intended target as it must modify its velocity vector in order to flock with subteam mates. On the other hand, if the scenario introduced blue subteams with more roles as in the previous paragraph it might be more important for blues to maintain spatial cohesion. Studies that examine the impact of flocking on blue's effectiveness may prove instructive.

Limitations in the simulator prevented using assignment frequency as an experimental factor; the simulator ran very slow at high frequencies. This was largely due to the Local Auction scenario in which the assignment algorithm had to be performed by every blue agent every time a reassignment was calculated. This led to longer simulation times as the number of blues in the simulator increased (a single simulation could run for up to seven hours at a reassignment frequency of 1 Hz). Distributed computing could alleviate this problem and assignment frequency could be varied and its impact on percentage of reds destroyed better investigated.

Another task assignment algorithm that might be considered for study but which requires some modification to our scenario is a simple relaxation algorithm in which some or all blues are always aloft (or on patrol) and assigned to reds at random when red subswarms arrive. Random pairs of blues periodically compare their utilities (e.g., distance to target or expected time of engagement) and exchange red targets if necessary. Simulated annealing [12] might also be employed to avoid local minima. It is likely that this approach would be less computationally complex that the auction approach presented herein and may be just as effective.

5.4 Lessons Learned

At the outset of the study we had intended to model agents in much greater fidelity. We had also intended to validate aspects of the simulator and prediction model via field experimentation. Lessons learned from attempts to realize those goals are here presented.

5.4.1 Simulating Multi-Agent Systems

At the outset of the experiment, the author made the goal to have the simulated agents at high fidelity as compared to physical UAVs. This was carried to the point that source code for control of both agent and physical UAV would be close to identical. The idea was that once the simulator was in place, any changes to the agent could quickly be ported to physical UAV software. Therefore, agents were not modeled in Repast Symphony, rather in Robot Operating System (ROS) [56] and a network service was created to allow data to pass between ROS agents and Repast Symphony. ROS would house the agents and Repast would provide simulation capability. During field experimentation, ROS-driven UAVs would operate on the same agent code developed during simulation.

Agents, however, did not scale to more than a few instances on a single machine. This issue arose simply because we were modeling at too high a fidelity. Each agent required its own process and behaved as though it had the use of the entire machine (as it would on a physical

UAV), adding overhead for each new agent. Once a machine housing ROS agents reached around 20 agents a machine was slowed significantly. The ability to simulate many agents was therefore resource intensive under this architecture.

Sensor simulation and radio communication simulation became another large problem. In order to simulate sensors, each agent needs to be able to quickly query other agents' positions – and each agent was housed in a different process and potentially on a different machine. In a non-simulated environment this is not necessary at all since UAVs simply read their sensor inputs. In many simulators, such as Repast Symphony, querying across processes and machines is not necessary because agents can query the simulator itself about the location of other agents with a nearly instantaneous response to the query. This quick response was not possible with agents modeled in ROS. A ROS agent communicates with other agents via a peer-to-peer network that does not broadcast or multicast communications. Recall that each ROS agent is operating in its own process and sharing processor time with all other ROS agents on that machine. In order for an agent to query other agents a message must be sent to each of the other agents. The processor then gives time to each of the other agents in their turn. As the number of ROS agents increases, the number of query messages increases much more quickly and the queries are not able to return data in a timely manner. This is still a problem when an oracle agent is used which stores positions of all nodes and is a central locations for queries to occur.

Due the previous problems, the simulator could only run at real time or slower. This did not allow for a large number of experiments to be done with diverse random inputs (Monte Carlo experimentation), which was one of the major goals of the simulator.

All of the computational and communication efficiency concerns are allayed by modeling the UAVs as simpler agents. Removing the requirement that physical UAVs have the same code as the simulated UAVs greatly simplifies the agent models and makes them much more scalable (thousands of agents on single machine). Agents were therefore modeled in Repast Symphony not ROS, and field experimentation has been left to future work. Repast models agents as lightweight threads that can communicate very efficiently with one another.

THIS PAGE INTENTIONALLY LEFT BLANK

REFERENCES

[1] J. R. Clapper *et al.*, "Office of the Secretary of Defense Unmanned Systems Roadmap (2009-2034)," United States Department of Defense, Tech. Rep., 2009.

[2] Jane's Unmanned Aerial Vehicles and Targets, "HESA Karrar," Available: http://www4.janes.com/subscribe/juav/doc_view.jsp?K2DocKey=/content1/janesdata/binder/juav/juava500.htm@current&Prod_Name=JUAV, September 2010.

[3] R. W. Robinson *et al.*, "2004 Report to Congress of the U.S.-China Economic and Security Review Commission," 108th United States Congress, 2nd Session, June 2004.

[4] J. S. Carson, "Introduction to Modeling and Simulation," in *Proc. of the 2005 Winter Simulation Conference*, Orlando, Florida, 2005, pp. 9–16.

[5] T. Fukuda *et al.*, "Self Organizing Robots Based on Cell Structures – CEBOT," in *Proc. of the IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS)*, Victoria, B.C., Canada, 1988, pp. 145–150.

[6] H. Guo *et al.*, "Swarm robot pattern formation using a morphogenetic multi-cellular based self-organizing algorithm," in *Proc. of the IEEE Int. Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011, pp. 3205–3210.

[7] G. Prencipe and N. Santoro, "Distributed Algorithms for Autonomous Mobile Robots," in *Proc. of the 4th IFIP Int. Conference on Theoretical Computer Science*, vol. 209, 2006, pp. 47–62.

[8] J. S. Bellingham *et al.*, "Multi-task assignment and path planning for cooperating uavs," in *Proc. of the Conference on Cooperative Control and Optimization*, Gainesville, Florida, November 2001, pp. 1–19.

[9] Y. Jin *et al.*, "Cooperative Real-time Search and Task Allocation in UAV Teams," in *Proc. of the 42nd IEEE Conference on Decision and Control*, vol. 1, December 2003, pp. 7–12.

[10] V. K. Shetty *et al.*, "Priority-based assignment and routing of a fleet of unmanned combat aerial vehicles," *Computers and Operations Research*, vol. 35, no. 6, pp. 1813–1828, June 2008.

[11] M. F. Munoz, "Agent-Based Simulation and Analysis of a Defensive UAV Swarm Against an Enemy UAV Swarm," Master's thesis, Naval Postgraduate School, June 2011.

[12] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2003.

[13] C. Schumaker *et al.*, "Task Allocation for Wide Area Search Munitions with Variable Path Length," in *Proc. of the 2003 American Control Conference*, Denver, CO, June 2003, pp. 3472–3477.

[14] K. S. Decker and V. R. Lesser, "Designing a Family of Coordination Algorithms," in *Proc. of the 1st Int. Conference on Multiagent Systems*, San Francisco, CA, 1995, pp. 32–51.

[15] Jane's Fighting Ships, "Arleigh Burke (Flight IIA) class," Available: http://www4.janes.com/subscribe/jfs/doc_view.jsp?K2DocKey=/content1/janesdata/yb/jfs/jfs_3533.htm@current&Prod_Name=JFS&QueryText=, August 2011.

[16] D. Johnson, "Rurhrstahl/kramer x-4," Available: http://www.luft46.com/missile/x-4.html, 2010.

[17] Sinodefence.com, "Pili-5 short-range air-to-air missile," Available: http://www.sinodefence.com/airforce/weapon/pl5.asp, October 2008.

[18] Army Recognition Magazine, "Crotale low-altitude surface-to-air missile system," Available: http://www.armyrecognition.com/france_french_army_vehicle_missile_systems_uk/crotale_low_altitude_ground_to_air_missile_system_technical_data_sheet_specifications_information_uk.html, 2008.

[19] M. Radomski, "Kill probability in antiaircraft firing theory," *Journal of Aircraft*, vol. 32, no. 6, pp. 1396–1399, 1995.

[20] Jane's Unmanned Aerial Vehicles and Targets, "IAI Harpy and Cutlass," Available: http://www4.janes.com/subscribe/juav/doc_view.jsp?K2DocKey=/content1/janesdata/binder/juav/juav0988.htm@current&Prod_Name=JUAV, December 2011.

[21] Jane's Unmanned Aerial Vehicles and Targets, "CIRPAS Pelican," Available: http://www4.janes.com/subscribe/juav/doc_view.jsp?K2DocKey=/content1/janesdata/binder/juav/juav9042.htm@current&Prod_Name=JUAV, December 2011.

[22] Jane's Unmanned Aerial Vehicles and Targets, "Unmanned Systems Orca," Available: http://www4.janes.com/subscribe/juav/doc_view.jsp?K2DocKey=/content1/janesdata/binder/juav/juava565.htm@current&Prod_Name=JUAV, February 2012.

[23] Jane's Unmanned Aerial Vehicles and Targets, "Unmanned Systems CT-450 Discoverer I," Available: http://www4.janes.com/subscribe/juav/doc_view.jsp?K2DocKey=/content1/janesdata/binder/juav/juava564.htm@current&Prod_Name=JUAV, February 2012.

[24] S. M. Ross, *Introduction to Probability Models*, 10th ed. Burlington, MA: Elsevier Inc., 2010.

[25] C. W. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," in *Proc. of the 14th Annual Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, Anaheim, California, July 1987, pp. 25–34.

[26] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, March 2006.

[27] D. Lee and M. W. Spong, "Stable flocking of multiple inertial agents on balanced graphs," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1469–1475, August 2007.

[28] N. Xiong *et al.*, "A survey on decentralized flocking schemes for a set of autonomous mobile robots," *Journal of Communications*, vol. 5, no. 1, pp. 31–38, January 2010.

[29] S. Fujita and V. R. Lesser, "Centralized task distribution in the presence of uncertainty and time deadlines," in *Proc. of the 2nd Int. Conference on Multiagent Systems*, Kyoto, Japan, December 1996, pp. 87–94.

[30] N. Miyata *et al.*, "Cooperative transport by multiple mobile robots in unknown static environments associated with real-time task assignment," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 767–780, October 2002.

[31] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, September 2004.

[32] M. Alighanbari and J. P. How, "Decentralized Task Assignment for Unmanned Aerial Vehicles," in *Proc. of the 44th IEEE Conference on Decision and Control and European Control Conference*, Seville, Spain, December 2005, pp. 5668–5673.

[33] M. Alighanbari and J. P. How, "Robust decentralized task assignment for cooperative UAVs," in *Proc. of the AIAA Conference on Guidance, Navigation, and Control*, Keystone, Colorado, August 2006.

[34] M. Alighanbari, "Robust and decentralized task assignment algorithms for uavs," Ph.D. dissertation, Massachusetts Institute of Technology, July 2007.

[35] R. Davis and R. G. Smith, "Negotiation as a Metaphor for Distributed Problem Solving," *Artificial Intelligence*, vol. 20, pp. 63–109, 1983.

[36] Y. Cai *et al.*, "Global Planning from Local Eyeshot: An Implementation of Observation-Based Plan Coordination in RoboCup Simulation Games," in *RoboCup 2001*, 2002, pp. 27–70.

[37] R. W. Beard and V. Stepanyan, "Information Consensus in Distributed Multiple Vehicle Coordinated Control," in *Proc. of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, USA, December 2003.

[38] R. Olfati-Saber and R. M. Murray, "Consensus Problems in Networks of Agents with Switching Topology and Time-Delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, September 2004.

[39] M. A. Olson and D. P. Porter, "An experimental examination into the design of decentralized methods to solve the assignment problem with and without money," *Economic Theory*, vol. 4, no. 1, pp. 11–40, January 1994.

[40] B. P. Gerkey and M. J. Matarić, "Sold!: Auction methods for multirobot coordination," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, 2002.

[41] N. Michael *et al.*, "Distributed multi-robot task assignment and formation control," in *Proc. of the IEEE Int. Conference on Robotics and Automation*, Pasadena, California, May 2008.

[42] B. J. Moore and K. M. Passino, "Distributed task assignment for mobile agents," *IEEE Transactions on Automatic Control*, vol. 52, no. 4, pp. 749–753, April 2007.

[43] Argonne National Laboratory, "Repast Simphony Agent Toolkit," Available: http://repast.sourceforge.net/index.html.

[44] W. L. Brogan, "Algorithm for ranked assignments with applications to multiobject tracking," *Journal of Guidance, Control, and Dynamics*, vol. 12, no. 3, pp. 357–364, May-June 1989.

[45] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization*, 1st ed. Belmont, MA: Athena Scientific, 1997.

[46] R. E. Burkard *et al.*, *Assignment Problems*, 1st ed. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2009.

[47] P. R. Halmos and H. E. Vaughan, "The Marriage Problem," *American Journal of Mathematics*, vol. 72, no. 1, pp. 214–215, January 1950.

[48] K. G. Murty, *Operations Research: Deterministic Optimization Models*, 1st ed. Englewood Cliffs, NJ: Prentice Hall, Inc., 1995.

[49] T. Shima *et al.*, "Multiple task assignments for cooperating uninhabited aerial vehicles using genetic algorithms," *Computers & Operations Research*, vol. 33, no. 11, pp. 3252–3269, November 2006.

[50] D. P. Bertsekas, *Network Optimization: Continuous and Discrete Models*, 1st ed. Belmont, MA: Athena Scientific, 1998.

[51] "GLPK (GNU Linear Programming Kit)," Available: http://www.gnu.org/software/glpk, October 2010.

[52] D. C. Montgomery, *Design and Analysis of Experiments*, 7th ed. Hoboken, NJ: John Wiley & Sons, Inc., 2009.

[53] H. O. Nyongesa *et al.*, "Genetic programming for anti-air missile proximity fuze delay-time algorithms," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 16, no. 1, pp. 41–45, January 2001.

[54] M. J. Muratore, "Effective teaming of airborne and ground assets for surveillance and interdiction," Master's thesis, U.S. Naval Postgraduate School, June 2010.

[55] SAS Institute Inc., "JMP ®, Version 9," Available: http://www.jmp.com/, Cary, North Carolina, 1989-2007.

[56] Willow Garage, “ROS.org (Robot Operating System),” Available: http://www.ros.org/wiki/.

APPENDIX 6:
Appendix

6.1 GLPK Model File For Transportation Problem

```
set R; /*reds*/
set B; /*blues*/

param D{i in R, j in B};
var X{R, B}, integer, >= 0;

minimize distances: sum{i in R, j in B} D[i,j] * X[i,j];

s.t. oneBAssignedToEachR{j in B}: sum{i in R} X[i, j] = 1; #EXACTLY one per (blues can only be assigned to a single red)
s.t. oneRAssignedToEachB{i in R}: sum{j in B} X[i, j] >= 3; #Each reds demands at least 3 blues

data;

#indices of reds
set R := 1 2 3 4 5 6 ;
#indices of blues
set B := 1 2 3 4 5 6 ;

#6 x 6 euclidean distance matrix
  param D : 1 2 3 4 5 6 :=
  1 627.181495905 566.380212365 677.63626389 319.463499152 582.451343117 989.122482157
  2 511.662736982 841.933103937 381.186580822 111.892799472 397.152694222 174.130135471
  3 943.348905814 696.51550162 280.34696968 474.697409718 597.297712715 55 6.24896907
  4 978.835242592 771.084851534 719.426512315 504.754066085 652.320096538 947.774497755
  5 724.437101479 687.018945928 39.4708262044 997.315396973 334.379303748 771.341542999
  6 888.85185317 853.966460952 883.633502551 779.288368983 851.255294101 8 58.596590958 ;

end;
```

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia
2. Dudly Knox Library
   Naval Postgraduate School
   Monterey, California
3. Dr. Timothy H. Chung
   Naval Postgraduate School
   Monterey, California
4. Dr. Chris Darken
   Naval Postgraduate School
   Monterey, California